MITSUBISHI

PROGRAMMABLE CONTROLLER

MELSEC-A

Programming Manual

type ACPU

MITSUBISHI ELECTRIC

REVISIONS

※The manual number is given on the bottom left of the back cover.

Print Date	Manual number	Revision
Sep., 1988	IB (NA) 66147-A	First edition

INTRODUCTION

Thank you for choosing the Mitsubishi MELSEC-A Series of General Purpose Programmable Controllers. Please read this manual carefully so that the equipment is used to its optimum. A copy of this manual should be forwarded to the end User.

CONTETNS

1. INTRODUCTION

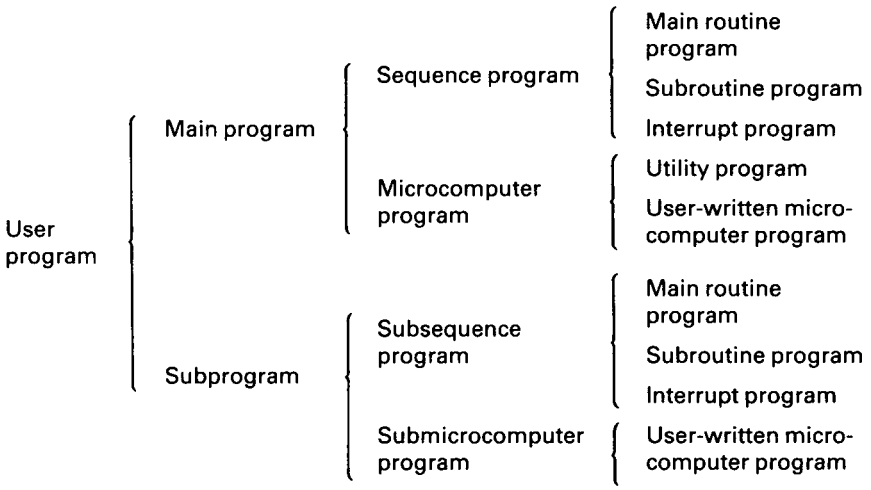
This manual describes performances, functions, instructions, etc. required for programming the MELSEC-A series programmable controllers.

This manual applies to the following CPU models:
A1N, A2N, A3NCPU
A3HCPU

"ACPU" in this manual indicates that the corresponding item applies to any of the above CPUs.

The MELSEC-A series PC parameters are used to specify the required functions, device ranges, etc.
Default value may be used as they are or may be changed as required.

The MELSEC-A series PC programs are classified as indicated below. Subprograms may only be written with the A3N, and A3HCPU.



IB (NA) 66147-A

1.1 Performance List

Item \ Type		A1NCPU	A2NCPU
Operation system		Stored program, repeated operation	
I/O control system		Refresh/direct mode selected	
Programming language		Language dedicated to sequence control (Relay symbol language, logic symbolic language, MELSAP language)	
Number of instructions	Sequence instruction	22	
	Basic instruction	131	
	Application instruction	101	105
Processing speed (Sequence instruction) (μ sec/step)		Direct mode: 1.0 to 2.3 Refresh mode: 1.0	
Number of I/O (point)		256	512
Watch dog timer (WDT)		Can be set in 10ms increments between 10 and 2000ms.	
Memory capacity (byte)		16K max.	Same as memory cassette capacity
Program capacity	Main sequence program (step)	6K max.	14K max.
	Subsequence program (step)	None	
	Microcomputer program (byte)	10K max.	26K max.
Number of internal relays (M) (point)		1000 (M0 to 999)	M+L+S=2048 (Set in parameters)
Number of latch relays (L) (point)		1048 (L1000 to 2047)	
Number of step relays (S) (point)		Depends on parameter setting (Defaults to 0)	
Number of link relays (B) (point)		1024 (B0 to 3FF)	
Timer (T)	Number of points	256	
	Specifications (s)	100ms timer: 0.1 to 3276.7 setting time (T0 to 199) 10ms timer: 0.01 to 327.67 setting time (T200 to 255) 100ms retentive timer: 0.1 to 3276.7 setting time } Set in parameters	
Counter (C)	Number of points	256	
	Specifications	Normal counter: 1 to 32767 setting range (C0 to 255) Interrupt counter (used in interrupt program): 1 to 32767 setting range } Set in parameters	
Number of data registers (D) (point)		1024 (D0 to 1023)	
Number of link registers (W) (point)		1024 (W0 to 3FF)	
Number of annunciators (F) (point)		256 (F0 to 255)	
Number of file registers (R) (point)		None	4096 max. (R0 to 4095)
Number of accumulators (A) (point)		2 (A0, A1)	
Number of index registers (V, Z) (point)		2 (V, Z)	
Number of pointers (P) (point)		256 (P0 to 255)	
Number of interrupt pointers (I) (point)		32 (I0 to 31)	
Number of special relays (M) (point)		256 (M9000 to 9255)	
Number of special registers (D) (point)		256 (D9000 to 9255)	
Number of comments (point)		128 (F0 to 127 used exclusively)	4032 max. (Set in units of 64 points)
Self-diagnostics		Watch dog error monitor, memory error detection, CPU error detection, I/O error detection, battery error detection, etc.	
Allowable instantaneous power failure time (ms)		20	

Table 1.1 Performance List (Continue)

A3NCPU	A3HCPU	Remarks
Stored program, repeated operation		
Refresh/direct mode selected		
Language dedicated to sequence control (Relay symbol language, logic symbolic language, MELSAP language)	Language dedicated to sequence control (Relay symbol language, logic symbolic language)	
22		
132		
109	107	
Direct mode: 1.0 to 2.3 Refresh mode: 1.0	Direct mode: 0.2 to 2 Refresh mode: 0.2 to 0.4	
2048		
Can be set in 10ms increments between 10 and 2000ms.	200ms only	
Same as memory cassette capacity		
30K max.		
30K max.		
58K max. for main and subprograms		
1000 (M0 to 999) 1048 (L1000 to 2047) Depends on parameter setting (Defaults to 0)	M+L+S=2048 (Set in parameters)	
1024 (B0 to 3FF)		
256		
100ms timer: 0.1 to 3276.7 setting time (T0 to 199) 10ms timer: 0.01 to 327.67 setting time (T200 to 255) 100ms retentive timer: 0.1 to 3276.7 setting time } Set in parameters		
256		
Normal counter: 1 to 32767 setting range (C0 to 255) Interrupt counter (used in interrupt program): 1 to 32767 setting range } Set in parameters	Normal counter: 1 to 32767 setting range (C0 to 255) Interrupt counter (used to count the number of interrupts): 1 to 32767 setting range (C224 to 255 may be used)	
1024 (D0 to 1023)		
1024 (W0 to 3FF)		
256 (F0 to 255)		
8192 max. (R0 to 8191)		
2 (A0, A1)		
2 (V, Z)		
256 (P0 to 255)		
32 (I0 to 31)		
256 (M9000 to 9255)		
256 (D9000 to 9255)		
4032 max. (Set in units of 64 points)		
Watch dog error monitor, memory error detection, CPU error detection, I/O error detection, battery error detection, etc.		
20		

Table 1.1 Performance List

1.2 Function List

Item \ Type		A1NCPU	A2NCPU
Status latch (byte)	Memory capacity	None	16K max.
	Data memory		None or 8K max. selected
	File register		None or 8K max. selected
Sampling trace	Memory capacity (byte)	None	8K max.
	Device setting		Device number
	Execution condition		Per scan or per hour selected
	Sampling count (number of times)		1024 max. (128 increments)
Offline switch memory (point)		256 for Y 2048 for M, L, S 1024 for B 256 for F	512 for Y 2048 for M, L, S 1024 for B 256 for F
Remote run/pause		None or X0 to FF selected	None or X0 to 1FF selected
Operation mode at the time of error		Stop/run selected	
Output mode switching from STOP to RUN		Output data at the time of STOP restored/output after operation execution selected	
Constant scan (ms) (Program execution at given time intervals)		May be set 10 increments between 10 and 2000.	
Latch range setting		Allowed for B0 to 3FF, T0 to 255, C0 to 255, D0 to 1023, W0 to 3FF	
Step run		Break point setting and run per instruction can be executed.	
Clock		Year, month, day, hour, minute, second, and day of the week can be written to and read from the special register.	

Table 1.2 Function List (Continue)

A3NCPU	A3HCPU	Remarks
24K max.		
None or 8K max. selected		
None or 16K max. selected		
8K max.		
Device number		
Per scan or per hour selected		
1024 max. (128 increments)		
2048 for Y 2048 for M, L, S 1024 for B 256 for F	2048 for Y 2048 for M, L 1024 for B 256 for F	
None or X0 to 7FF selected		
Stop/run selected		
Output data at the time of STOP restored/output after operation execution selected		
May be set in 10 increments between 10 and 2000.	May be set in 10 increments between 10 and 190.	
Allowed for B0 to 3FF, T0 to 255, C0 to 255, D0 to 1023, W0 to 3FF		
Break point setting and run per instruction can be executed.		
Year, month, day, hour, minute, second, and day of the week can be written to and read from the special register.		

Table 1.2 Function List

2. DESCRIPTION OF DEVICES

2.1 Device List

Items marked ✳ in the table are enabled, or their ranges assigned, in the PC parameters using the peripheral equipment. For details of parameter setting, see Section 3.5 "Parameter Setting."

Device \ Type			A1NCPU	A2NCPU	A3N, A3HCPU	Remarks
	X	Input	X, Y0 to FF (X + Y = 256 points)	X, Y0 to 1FF (X + Y = 512 points)	X, Y0 to 7FF (X + Y = 2048 points)	X, Y numbers are in hexadecimal.
	Y	Output				
	M	Special relay	M9000 to 9255 (256 points)			
✳	M	Internal relay	M0 to 999 (1000 points)			The function of step relays is identical to that of internal relays.
✳	L	Latch relay	L1000 to 2047 (1048 points)		(M + L + S = 2048 points)	
✳	S	Step relay	Set in parameters (Defaults to 0 point)			
	B	Link relay	B0 to 3FF (1024 points)			B numbers are in hexadecimal.
	F	Annunciator	F0 to 255 (256 points)			
✳	T	100ms timer	T0 to 199 (200 points)			There are 256 points of timers and counters, respectively.
✳	T	10ms timer	T200 to 255 (56 points)		(100ms + 10ms + 100ms retentive timers = 256 points)	
✳	T	100ms retentive timer	Set in parameters (Defaults to 0 point)			
✳	C	Counter	C0 to 255 (256 points) (Counters + interrupt counters = 256 points)			
✳	C	Interrupt counter	Set in parameters (Defaults to 0 point) Counter for use in interrupt program		Counter used to count the number of interrupt signals	
	D	Data register	D0 to 1023 (1024 points)			
	D	Special register	D9000 to 9255 (256 points)			
	W	Link register	W0 to 3FF (1024 points)			W numbers are in hexadecimal.
✳	R	File register	———	Set in parameters (Defaults to 0 point) 1 to 4K points	Set in parameters (Defaults to 0 point) 1 to 8K points	
	A	Accumulator	A0, A1 (2 points)			
	Z	Index register	Z (1 point)			
	V	Index register	V (1 point)			
	N	Nesting	N0 to 7 (8 levels)			
	P	Pointer	P0 to 255 (256 points)			
	I	Interrupt pointer	I0 to 31 (32 points)			
	K	Decimal constant	K−32768 to 32767 (16 bit instruction)			
	K	Decimal constant	K−214783648 to 214783647 (32 bit instruction)			
	H	Hexadecimal constant	H0 to FFFF (16 bit instruction)			
	H	Hexadecimal constant	H0 to FFFFFFFF (32 bit instruction)			

Table 2.1 Device List

IB (NA) 66147-A

2.2 Input/Output X, Y

Via the inputs and outputs, communication is made between the PC and external equipment.
Inputs are used to receive ON/OFF data for use in the program from external devices to the input modules. Outputs are used to provide program operation results from the output modules to the external devices.

(1) Input X

1) Inputs give commands and data from external devices (e.g. pushbuttons, select switches, limit switches, digital switches) to the PC.

2) Regarding that one point of input incorporates a virtual relay Xn in the PC, the N/O contact and N/C contact of that Xn are used in the program.

3) There is no restriction on the number of N/O contacts and N/C contacts of Xn used in the program.

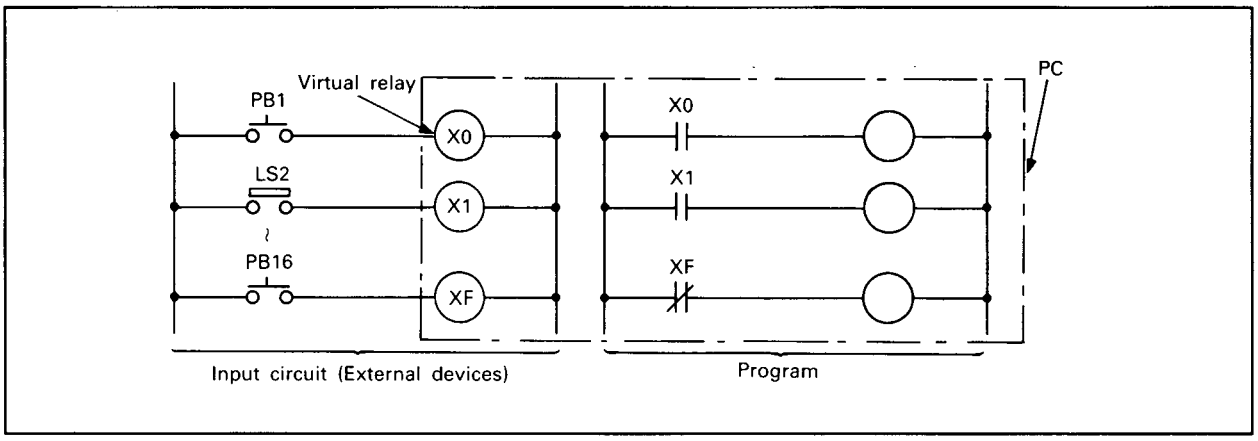


Fig. 2.1 Inputs (X)

(2) Output Y

1) Outputs provide program control results to external devices (e.g. solenoids, magnetic switches, signal lamps, digital indicators).

2) Outputs can be fetched to the outside as an equivalent to one N/O contact.

3) There is no restriction on the number of N/O contacts and N/C contacts of Yn used in the program.

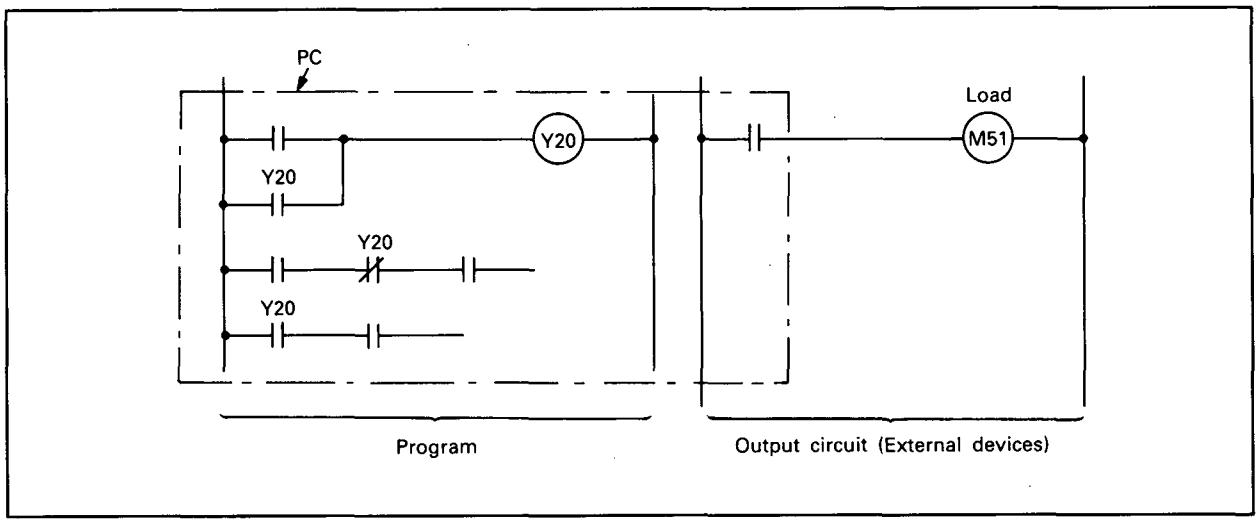


Fig. 2.2 Outputs (Y)

2.3 Internal Relay M, Latch Relay L, Step Relay S

Auxiliary relays for use in the PC. There is no limit to the number of contacts (NO, NC contacts) used in the program.

(1) Internal relay M, step relay S

1) Cannot be latched. Hence, all internal relays are switched off if the PC is switched on, reset or latch-cleared.

(2) Latch relay L

1) Battery backed, i.e. operation results are retained if the PC is switched on or reset.

2) Latch-clear to switch off the latch relays from the external device.

2.4 Link Relay B

1) Internal relay for use in a data link system.

2) The ON/OFF data of the link relays used in a data link system can be read by switching them on/off as coils in the host (master, local station) and as contacts in other stations (master, local stations). Link relays thus allow ON/OFF data to be transferred between the master and local stations.

3) The link range (range of link relays for use as coils in each station) must be set to the master station. Link relays outside the link range may be used as internal relays.

4) There is no restriction on the number of N/O contacts and N/C contacts of link relay used in the program.

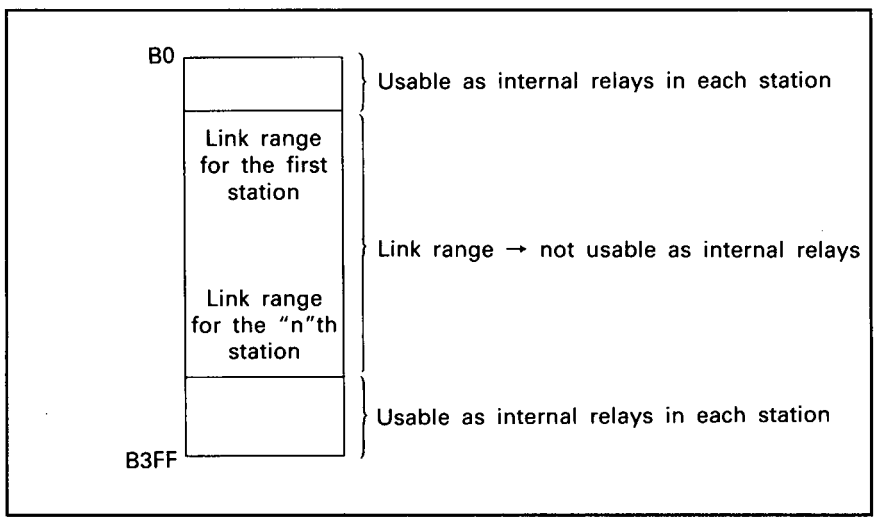


Fig. 2.3 Assignment of Link Relays

2.5 Annunciator F

1) Used to detect a fault. By writing a fault detection program using annunciators, annunciators are scanned by the fault detection program during run and the corresponding annunciator is switched on if a fault occurs.

2) The earliest annunciator (F) number detected is written to special register D9009.

3) F numbers detected are stored to registers D9125 to D9132 in the order in which they occur on a First In First Out basis.

4) The D9124 value is incremented by 1 each time any of F0 to 255 is switched on and is decremented by 1 each time the RSF F[ ] or LED instruction is executed (or by pressing the INDICATOR RESET switch on the A3N, A3HCPU front panel).

5) By resetting the F number in D9009,

   - The reset F number coil is switched off.
   - The next earliest F number detected, if any, is stored to D9009.

6) Execute the [RST] F[ ] instruction to reset the detected annunciator coil.

2.6 Timer T

Up-timing timer which begins timing when its coil is switched on and times out when the present value reaches the set value. The timer contacts close when the timer times out.

(1) 100ms, 10ms timers

1) The present value is reset to 0 and the contacts open when the coil is switched off.

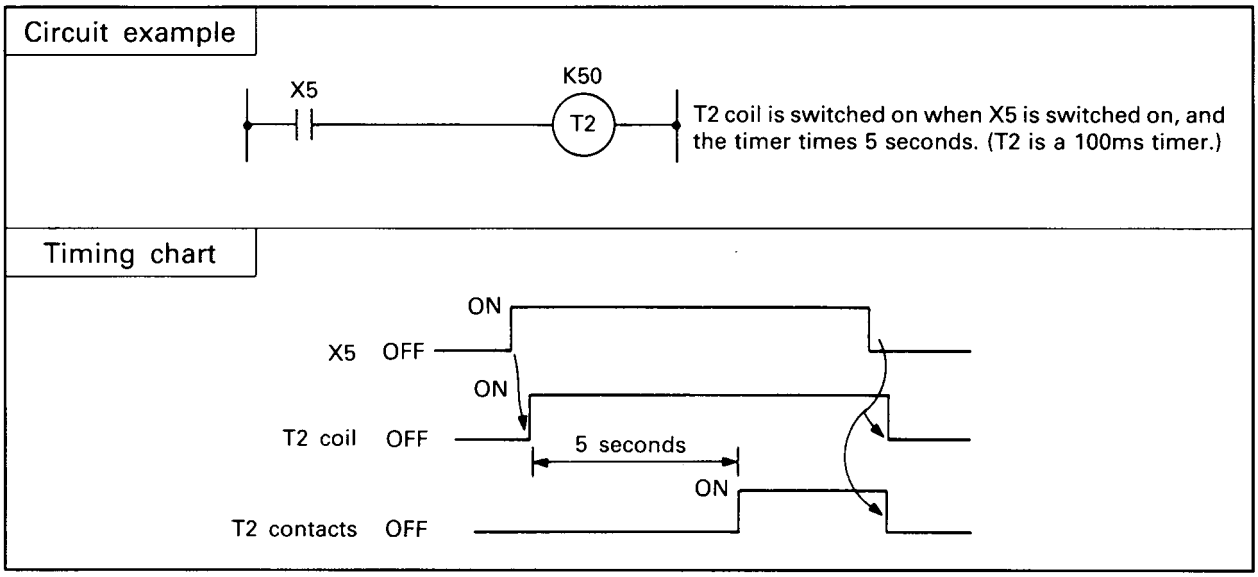


Fig. 2.4 Timing Chart

(2) 100ms retentive timer

1) Used to time the coil ON period. Therefore, the present value and contact status are retained if the coil is switched off. Timing is resumed at the retained present value when the coil is switched on.

2) Use the RST T ☐ instruction to clear the present value and open the contacts.

2.6.1 Timer processing and accuracy

1) With continuity in front of a timer coil, the timer present value and contact status are updated after the execution of the [END] (or FEND) instruction and the timer contacts close after the timer has timed out.

2) When the continuity is removed from in front of the timer coil, the present value is reset to 0 (retentive timers retain their present value and are reset using the [RST] command), and the timer contacts open.

3) The timer present value is cleared and the contacts open when the timer is reset by the [RST] instruction.

4) If a timer is jumped, as in the example below, after it has started timing, it continues to time even if the preceding continuity is removed. (This is because the PC is no longer scanning the jumped program area.)

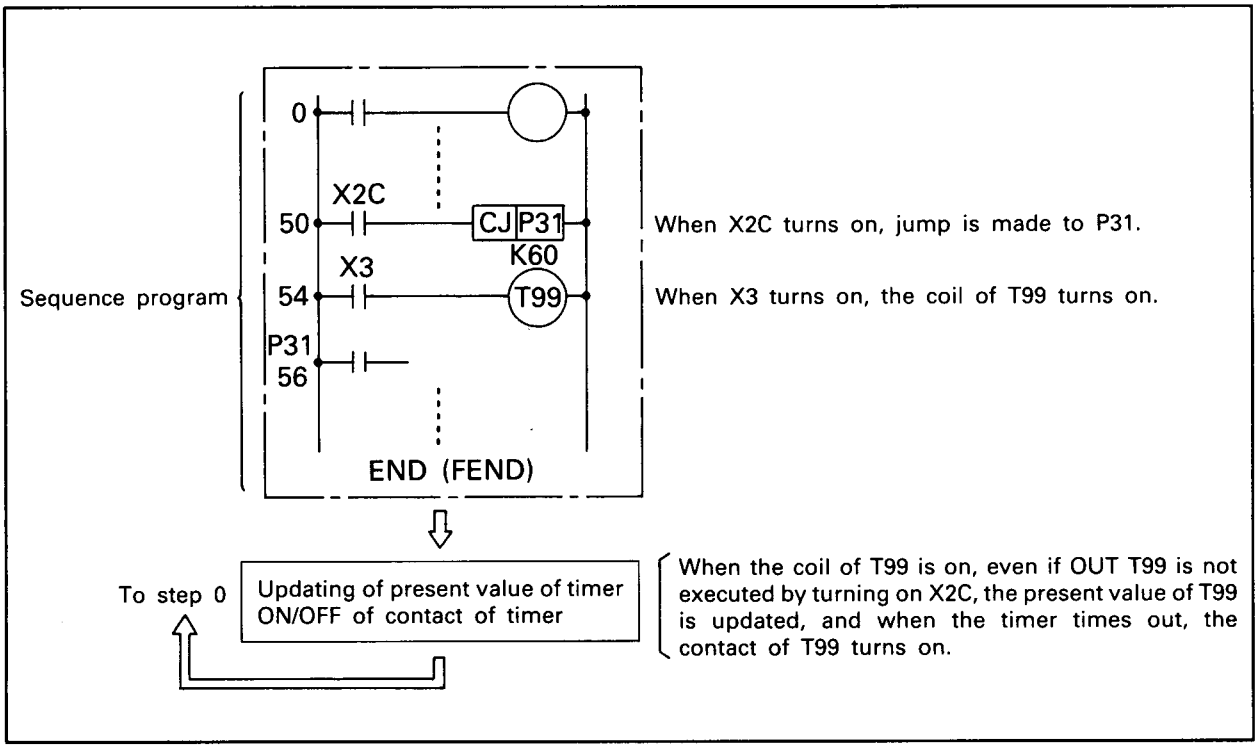


Fig. 2.5 Timer Processing

5) The timer present value update timing and accuracy in direct and refresh modes are explained on the following pages.
When the input (X) is used as a condition contact in front of the timer coil, accuracy differs between modes. For any other device used as a condition contact, see direct mode processing.

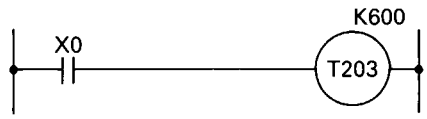
(1) Present value update timing and accuracy in direct mode

1) Timer accuracy depends on the timer and scan time.

Timer Type	Scan Time T	Accuracy
10ms	T < 10ms	+2 scan time to −10ms
10ms	T ≧ 10ms	+2 scan time to −1 scan time
100ms, 100ms retentive	T < 100ms	+2 scan time to −100ms
100ms, 100ms retentive	T ≧ 100ms	+2 scan time to −1 scan time

2) The following example indicates the present value update timing and accuracy with a 10ms timer used in the program of 10ms or more scan time.

Circuit example



When the timer measures 6 seconds after X0 turns on, the contact of T203 turns on. (T203 is a 10ms timer.)

Measuring method of timer

For scan time the 25ms

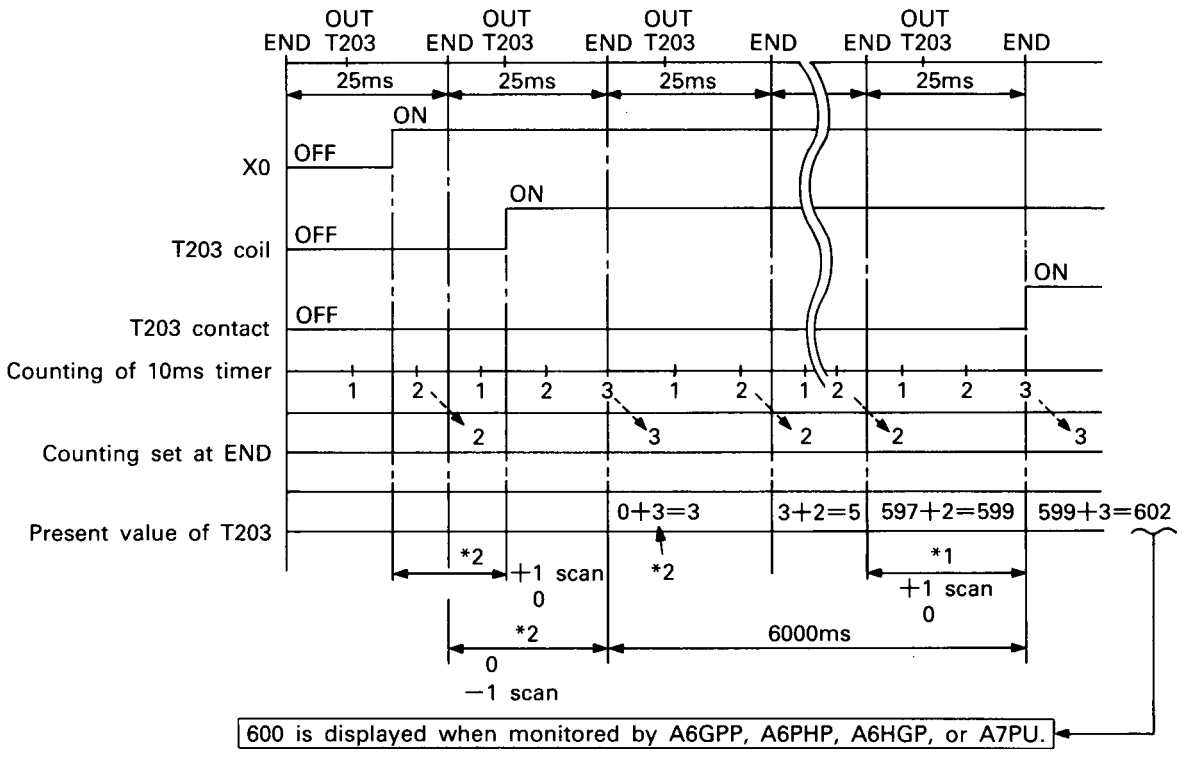


Fig. 2.6 Timer Timing

T203 time-out period includes the following errors:

*1: 10ms timer error (+1 scan time)
*2: Error depending on timing of timer input continuity and location of the OUT T⬚ instruction in program (±1 scan time)

Accuracy is therefore $^{+2 \text{ scan time}}_{-1 \text{ scan time}}$ ($^{+0.05}_{-0.025}$ seconds in Fig. 2.6).

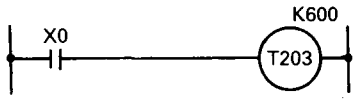
3) Contact status is updated only after the END instruction is processed, regardless of the timer coil status during any scan.

(2) Update timing and accuracy in refresh mode

1) Timer accuracy is +2 scan time independently of the timer and scan time.

2) The following example indicates the present value update timing and accuracy by using a 10ms timer in a program of 10ms or more scan time.

Ladder example



When the timer has measured 6 seconds after X0 enables, T203 contact turns on. (T203 is 10ms timer)

Timer measuring method

Scan time : 25ms

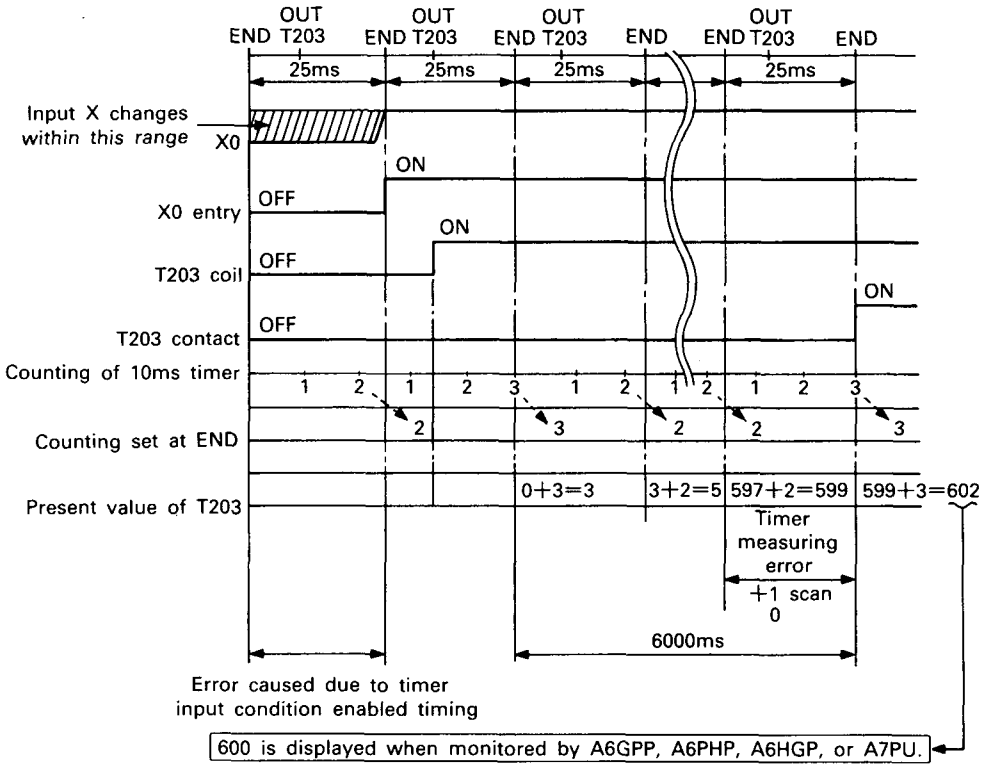


Fig. 2.7 Timer Timing

T203 time-out period includes the following errors:

*1: 10ms timer error (+1 scan time)

*2: Error depending on timing of timer input continuity and location of the OUT T□ instruction in program (±1 scan time)

Accuracy is therefore +2 scan time (+0.05 seconds in Fig. 2.7)

3) Contact status is updated only after the END instruction is processed, regardless of the timer coil status during any scan.

2.7 Counter C

1) Up counter which counts out when the count value reaches the set value.

2) The counter counts the leading edges of pulses driving its coil and counts once only when the coil is switched from off to on.

3) When the counter coil is switched on, the counter present value and contact status are updated after the [END] (FEND) instruction is executed.

4) The count value is not cleared if the coil is switched off. Use the [RST] C⬚ instruction to clear the count value and update the contact status.

Ladder example

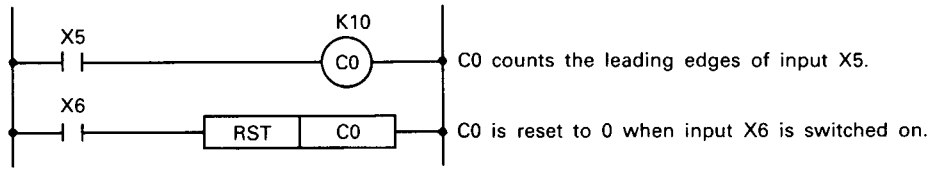


Fig. 2.8 Count Ladder

2.7.1 Maximum counting speed of normal counter

(1) Count value update timing in direct mode

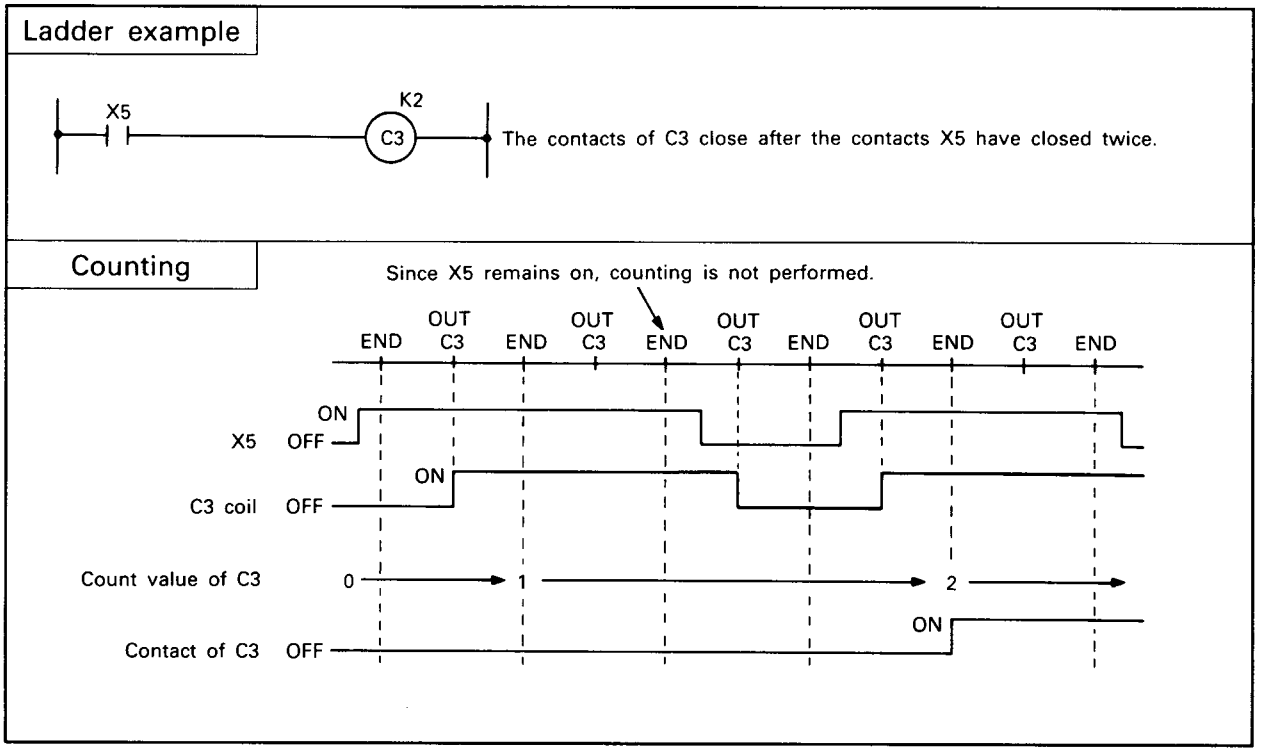


Fig. 2.9 Counter Counting

The maximum counting speed of the counter depends on the scan time. Counting is only possible if the input condition is on for more than one scan time.

$$\text{Maximum counting speed (Cmax.)} = \frac{n}{100} \times \frac{1}{ts} \text{ (times/sec)}$$

n: duty (%), For details, see Fig. 2.12.
ts: Program scan time (sec)

(2) Count value update timing in refresh mode

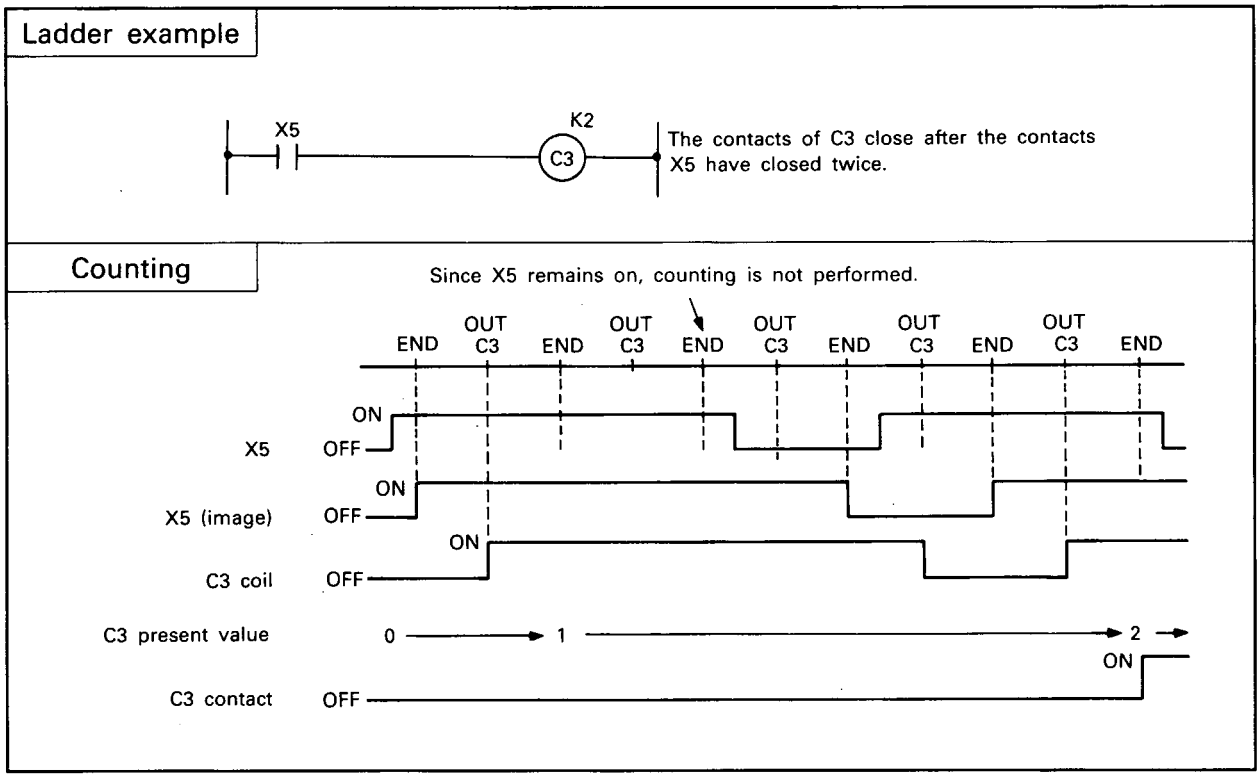


Fig. 2.10 Counter Counting

$$\text{Maximum counting speed (Cmax.)} = \frac{n}{100} \times \frac{1}{ts} \text{ (times/sec)}$$

n: duty (%), For details, see Fig. 2.12.
ts: Program scan time (sec)

2.8 Counter C for Interruption

Two types of interrupt counter are available. One is used in the interrupt program and the other counts the number of interrupt signals.
The interrupt program counter may be used with the A1N, A2N and A3NCPU. The interrupt signal counter may be used with the A3HCPU.

2.8.1 Interrupt program counter

1) When a counter coil is switched on, the counter present value and contact status are updated after the IRET instruction is executed.
A counter counts the leading edges of the pulses driving its coil and counts once only when its input condition changes from off to on.

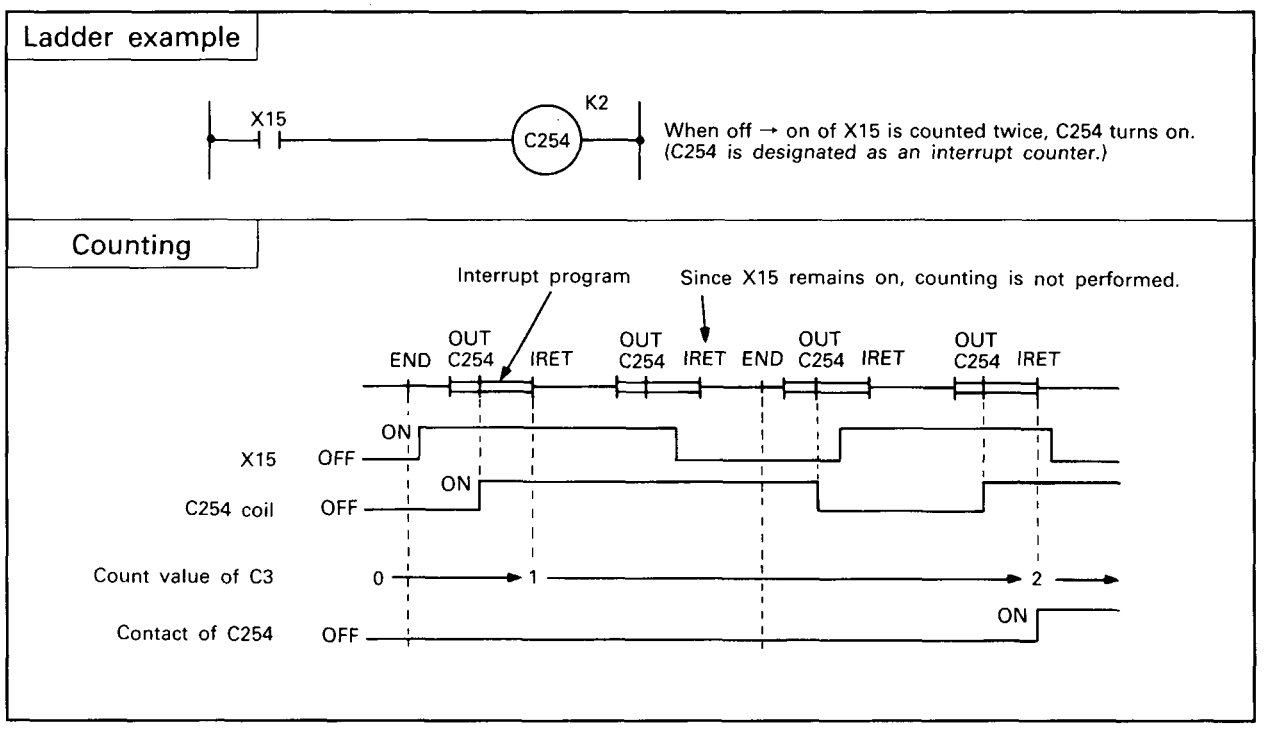


Fig. 2.11 Counting Method of Counter for Interruption

2) The maximum counting speed of the interrupt counter depends on the interrupt signal interval. Counting is only possible if the input condition is on for more than the interrupt signal interval.

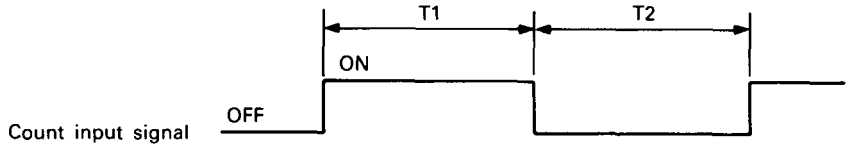
$$\text{Maximum counting speed (Cmax.)} = \frac{n}{100} \times \frac{1}{ti} \text{ (times/sec)}$$

where, n = duty (%)
ti = interrupt signal interval (sec)

Duty is the ratio of the input signal's on time to off time as a percentage.

$$\text{If } T1 \leqq T2 \quad n = \frac{T1}{T1 + T2} \times 100[\%]$$

$$\text{If } T1 > T2 \quad n = \frac{T2}{T1 + T2} \times 100[\%]$$



2.8.2 Interrupt signal counter

1) C244 to C255 may used as interrupt signal counters. The counter present value and contact status are updated when an interrupt occurs.

2) Interrupt pointers may assigned to counters as shown below. The corresponding counter counts when any interrupt relevant to the interrupt pointer occurs. For details of the interrupt pointers, see Section 2.16.

Interrupt Pointer	Interrupt Counter	Interrupt Pointer	Interrupt Counter	Interrupt Pointer	Interrupt Counter	Interrupt Pointer	Interrupt Counter
I0	C224	I8	C232	I16	C240	I24	C248
I1	C225	I9	C233	I17	C241	I25	C249
I2	C226	I10	C234	I18	C242	I26	C250
I3	C227	I11	C235	I19	C243	I27	C251
I4	C228	I12	C236	I20	C244	I28	C252
I5	C229	I13	C237	I21	C245	I29	C253
I6	C230	I14	C238	I22	C246	I30	C254
I7	C231	I15	C239	I23	C247	I31	C255

3) Counters C224 to C255 may be used as interrupt signal counters by setting parameters in 1 point increments.

4) An interrupt program cannot be written to an interrupt pointer (I) to which an interrupt counter has been allocated.

5) The interrupt signal counter does not count during any of the following:

- Instruction execution.
- END processing interrupt disable area (timer, counter present value update, etc.).
- Interrupt program processing.

6) The maximum counting speed can be calculated using the longest processing time of the following:

- Instruction with the longest processing time present in the program
- END processing interrupt disable area ····· max. 2ms
- Interrupt program processing time

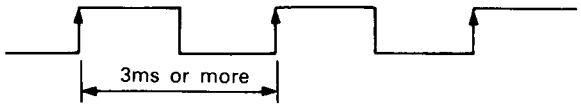
$$\text{Max. counting speed} = \frac{1}{(\text{max. processing time of the above}) + (500\ \mu\text{s} \times \text{number of interrupt counters})}\ (\text{PPS})$$

Example:

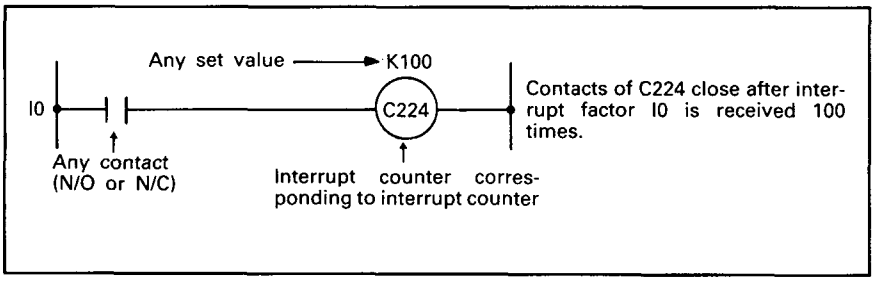
The END processing time is 2ms. If the max. instruction processing time is 0.3ms, a program is not written during run, there is no interrupt program, and two interrupt counters are used.

$$\text{Max. counting speed} = \frac{1}{0.002+0.0005\times 2} \fallingdotseq 333\ (\text{PPS})$$

Hence, the highest speed pulse train which may be reliably read by the A3HCPU, with the above conditions is 333 pulses/sec.



7) In order to use an interrupt counter, the following program rung must be written after the FEND and before the END instructions in addition to parameter setting.



Interrupt should be kept enabled by executing the EI instruction at the program head.

8) • The interrupt counter contact may be used at any location in the sequence program.
• The interrupt counter keeps counting after the count value reaches the set value.
• The count value can be transferred using the MOV instruction, etc. in the sequence program.
• The count value is reset to 0 by executing the RST instruction located before the FEND instruction in the sequence program.

9) Using many interrupt counters may increase the sequence program processing time and cause "WDT ERROR." In this case, the number of interrupt counters must be reduced or the input pulse counting speed lowered.

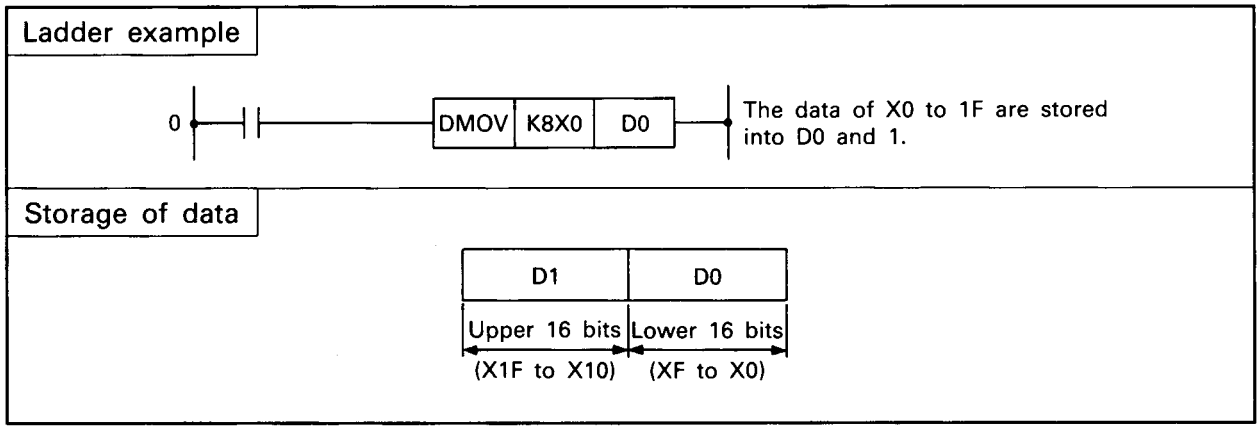
2.9 Data Register D

1) The data register is a memory which stores data inside the PC.

2) Data registers consist of 16 bits and allow read and write operations requiring 16 bits.

3) When 32-bit data is handled, two registers are used. The data register number specified by the 32-bit instruction contains the lower 16 bits and the specified data register number +1 contains the upper 16 bits.

Example: This example shows a circuit which uses the DMOV instruction.



4) The data stored by the sequence program is retained until other data is stored.

5) The data stored in the data register is cleared by turning on the power or moving the RESET switch to the "RESET" position.
The latched data registers are cleared by moving the RESET switch to the "LATCH CLEAR" position with the RUN key switch located at the STOP position.

6) The following devices can be used as data registers. (if not utilized)

- Unused timers (T) and counters (C)
- File registers (R) in the range specified by parameter setting
- Link registers (W) which are not used for data link
- Index registers (Z, V)
- Accumulators (A0, A1)

REMARKS

When the power failure compensation (latch) is required for the data registers, it can be set by the parameter setting of peripheral equipment.

2.10 Link Register W

1) Data register for use with the data link.

2) In a data link system, data is written to link registers at the host (master, local station) and read from link registers at the other stations (master, local stations). Link registers thus allow data to be transferred between the master and local stations.

3) Before using link registers, the link range must be set to the master station.
Link registers outside the link range may be used as data registers at each station.

4) Link registers consist of 16 bits and allow read and write operations requiring 16 bits. When 32-bit data is handled, two registers are used. The link register number specified by the 32-bit instruction contains the lower 16 bits and the specified link register number +1 contains the upper 16 bits.

5) The data stored in the link register is cleared by turning on the power or moving the RESET switch to the "RESET" position.
The latched link registers are cleared by moving the RESET switch to the "LATCH CLEAR" position with the RUN key switch located at the STOP position.

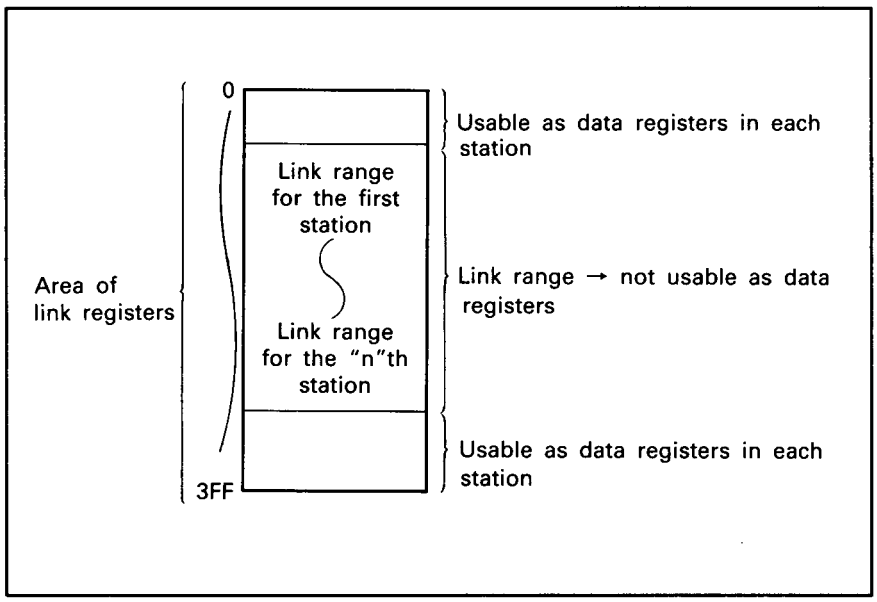


Fig. 2.12 Assignment of Link Registers

2.11 File Register R

1) Used as extra data registers in the user memory area of the memory cassette.

2) Like the data register, the file register can be used in the sequence program.

3) File registers consist of 16 bits and allow read and write operations requiring 16 bits. When 32-bit data is handled, two registers are used. The file register number specified by the 32-bit instruction contains the lower 16 bits and the specified file register number +1 contains the upper 16 bits.

4) Data stored in any file register remains unchanged if the power is switched off, and is not cleared if the power is switched on or the RESET switch is set to the "RESET" or "LATCH CLEAR" position with the RUN key switch located at STOP position.

5) To clear the file register, write 0 by use of the FMOV(P) instruction. (For the FMOV(P) instruction, refer to Fig. 6.4.3.)

Example: This example shows the clearing of R0 to 1023 (1K points).

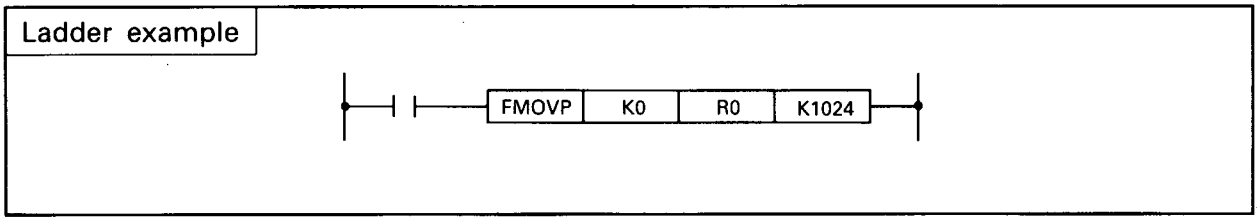


Fig. 2.13 File Register Clear Ladder Example

POINT

To use the file registers, it is required to set the number of file register points.

2.12 Accumulator A

1) The accumulator is a data register which stores the operation results of basic instructions and application instructions. Basic instructions and application instructions, of which operation results are stored, are as indicated below.

Instruction	Ref. page	Instruction	Ref. page	Instruction	Ref. page
SER	7-38 to 7-39	DROR	7-26 to 7-27	RCL	7-24 to 7-25
SERP		DRORP		RCLP	
SUM	7-40 to 7-41	RCR	7-22 to 7-23	DROL	7-28 to 7-29
SUMP		RCRP		DROLP	
DSUM	7-40 to 7-41	DRCR	7-26 to 7-27	DRCL	7-28 to 7-29
DSUMP		DRCRP		DRCLP	
ROR	7-22 to 7-23	ROL	7-24 to 7-25		
RORP		ROLP			

2) When an instruction other than the above is used, the accumulator can be used in the sequence program as a register equivalent to the data register.

3) The accumulator consists of 16 bits and allows read and write operations in units of 16 bits.

4) The accumulators are available in two points (A0, A1). In the 32-bit instruction, A0 stores the lower 16 bits and A1 stores the upper 16 bits. Therefore, A1N cannot be specified for the 32-bit instruction.

5) The contents of accumulator are cleared by turning on the power or moving the RESET switch on the front of CPU to the "RESET" position.
Accumulators are also cleared by setting the RESET switch to the "LATCH CLEAR" position with the RUN key switch located at the STOP position.

2.13 Index Registers Z, V

1) The index registers are used for the qualification of devices (X, Y, M, L, S, B, F, T, C, D, W, R, K, H, P).
Note that when used with any bit device, the index register may only be used to specify the digit.

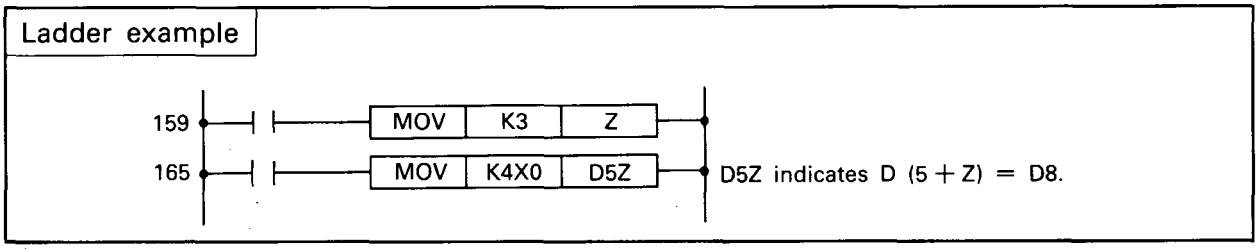


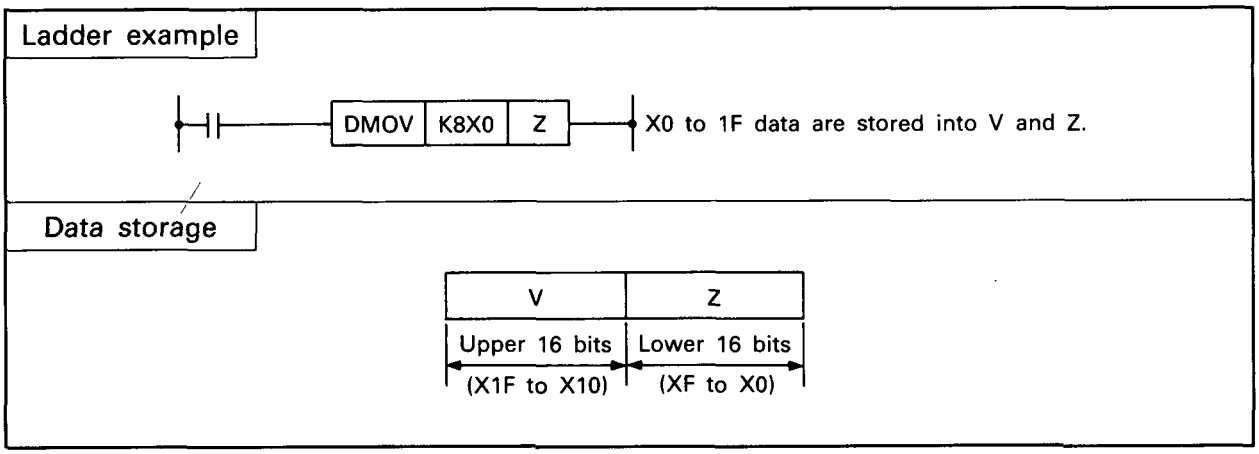
Fig. 2.14 Index Register Qualification Ladder

2) The index registers can be used for the sequence program like the data registers.

3) The index register is 1 point and consists of 16 bits. Write and read operations can be performed per 16 bits.

4) There are 2 points (Z, V) of index registers. In a 32-bit instruction, Z is lower 16 bits and V is upper 16 bits. Therefore, V cannot be specified by the 32-bit instruction.

Example: Indicates an example by use of the DMOV instruction.



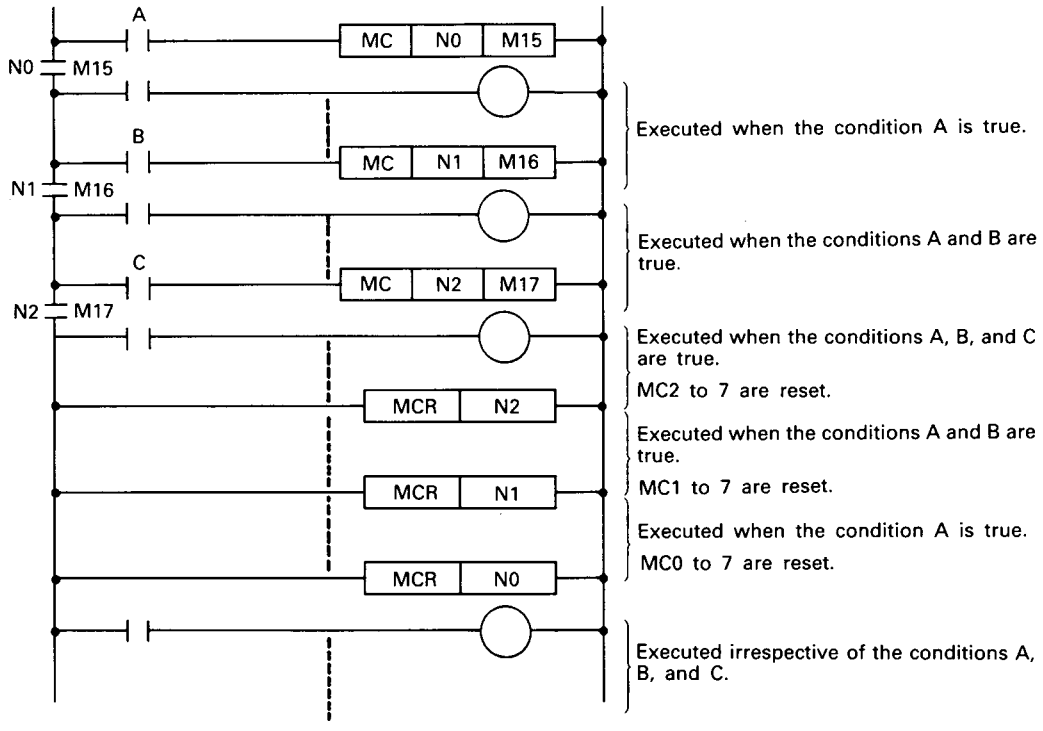
5) The contents of index registers are cleared when the power is turned on or by moving the RESET switch on the CPU front to the "RESET" position.
The index registers are also cleared by setting the RESET switch to the "LATCH CLEAR" position when the RUN key switch located at the STOP position.

2.14 Nesting N

1) A loop of master controls to several levels.

2) The MC to MCR ladder has various input conditions with contacts provided at the bus.

3) Nesting (N) numbers should be used in serial order.



① The condition, by which each master control turns on, is as indicated below.

MC N0 M15 ..... On when the condition A turns on.

MC N1 M16 ..... On when the condition A and B turn on.

MC N2 M17 ..... On when the conditions A, B, and C turn on.

Fig. 2.15 Master Control Nesting

4) When the master control is off, the states of timers and counters are as follows:

- 100ms timer, 10ms timer: Count value turns to 0.
- 100ms retentive timer: Remains at the present count value.
- Counter: Remains at the present count value.
- OUT instruction: All are turned off.

2.15 Pointer P

1) The pointer indicates the jump destination of branch instruction (CJ, SCJ, CALL, JMP) and the pointer number attached to the beginning of jump destination is referred to as a label.

2) The same label cannot be used multiple times. Multiple use results in error.

3) P255 always indicates [END].
(P255 can be used as a device of CJ, SCJ instruction, etc. However, it cannot be used as a label. Also, P255 cannot be used as the device of [CALL] instruction.)

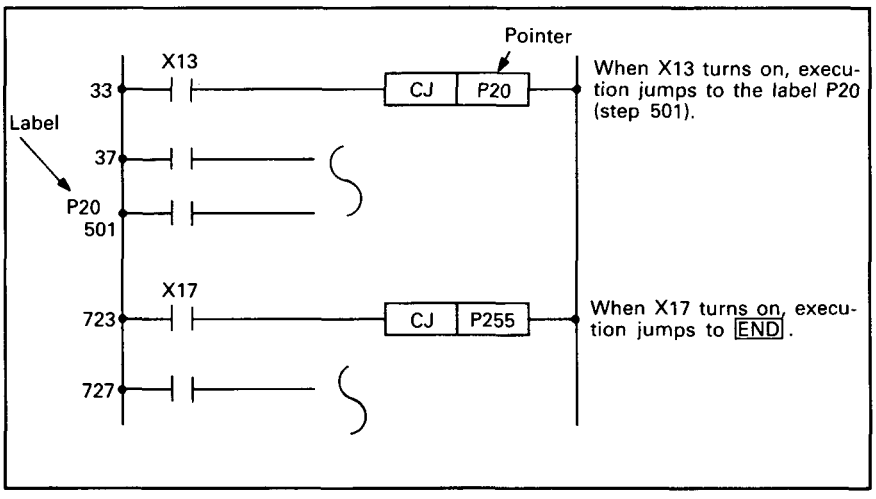


Fig. 2.16 Pointers Used with Branch Instructions

2.16 Pointer for Interruption I

1) When an interrupt factor occurs, the pointer for interruption indicates the jump destination to an interrupt program corresponding to the interrupt factor.

2) Provide the same label as the interrupt pointer at the head of interrupt program.

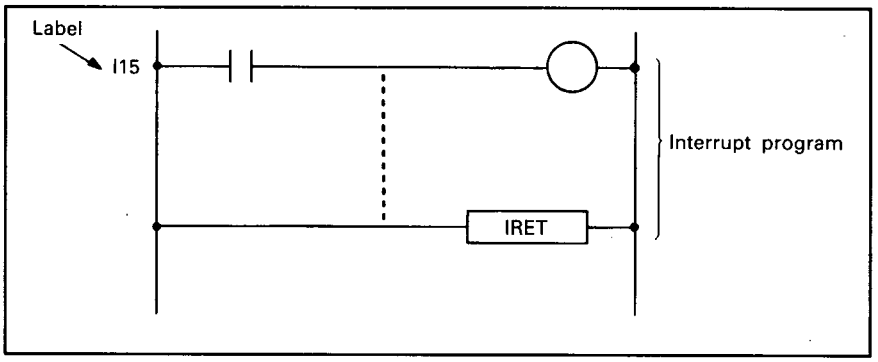
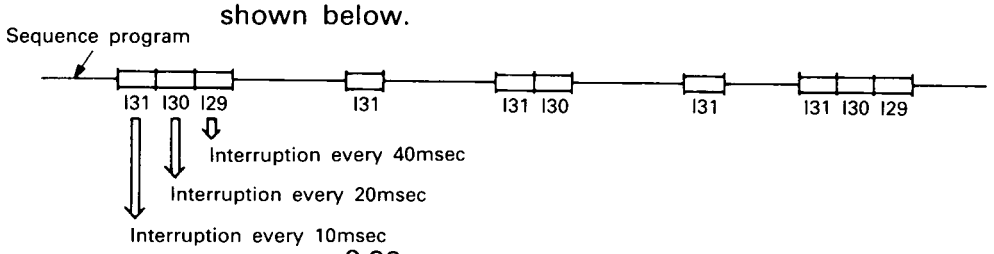


Fig. 2.17 Interrupt Pointer

3) The applications of pointers for interruption are as indicated below.

Priority		Pointer for Interruption	Interrupt Sequence Program Starting Factor
2	High ↓ Low	I0 I1 ≀ I15	Process interrupt unit (AI61) 1st point interruption 2nd point interruption 16th point interruption
1	High ↓ Low	I16 I17 ≀ I23	Sequence start occurrence unit 1st module 2nd module 8th module
—	—	I24 ≀ I28	Unusable
3	Low ↓ High	I29 I30 I31	Interruption per 40msec Interruption per 20msec Interruption per 10msec

4) When I29 to 31 exist in the program, jump is made to that interrupt program and the interrupt program is executed per interrupt time.
After all of I29 to 31 are used, the execution is performed as shown below.



2.17 Special Relay M

The special relay is an internal relay of which application has already been determined inside the PC. They must not be switched on/off in the program (except those marked *1, *2 in the table).

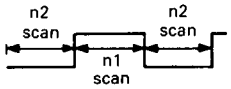
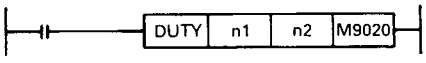
Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
*1 M9000	Fuse blown	OFF: Normal ON: Presence of fuse blow unit	●Turned on when there is one or more output units of which fuse has been blown. Remains on if normal status is restored.	●	●	●	●
*1 M9002	I/O unit verify error	OFF: Normal ON: Presence of error	●Turned on if the status of I/O module is different from entered status when power is turned on. Remains on if normal status is restored.	●	●	●	●
*1 M9005	AC DOWN detection	OFF: AC power good ON: AC power DOWN	●Turned on if power failure is greater than 10ms. Reset when POWER switch is moved from OFF to ON position.	●	●	●	●
*1 M9006	Battery low	OFF: Normal ON: Battery low	●Turned on when battery voltage reduces to less than specified. Turned off when battery voltage becomes normal.	●	●	●	●
*1 M9007	Battery low latch	OFF: Normal ON: Battery low	●Turned on when battery voltage reduces to less than specified. Remains on if battery voltage becomes normal.	●	●	●	●
*1 M9008	Self-diagnostic error	OFF: Absence of error ON: Presence of error	●Turned on when error is found as a result of self-diagnosis.	●	●	●	●
M9009	Annunciator detection	OFF: No detection ON: Detection present	●Turned on when [OUT] F or [SET] F instruction is executed. Switched off when D9124 data is zeroed.	●	●	●	●
M9010	Operation error flag	OFF: Absence of error ON: Presence of error	●Turned on when operation error occurs during execution of application instruction. Turned off when error is eliminated.	●	●	●	–
*1 M9011	Operation error flag	OFF: Absence of error ON: Presence of error	●Turned on when operation error occurs during execution of application instruction. Remains on if normal status is restored.	●	●	●	●
M9012	Carry flag	OFF: Carry off ON: Carry on	●Carry flag used in application instruction.	●	●	●	●
M9016	Data memory clear flag	OFF: No processing ON: Output clear	●Clears the data memory including the latch range (other than special relays and special registers) in remote run mode from computer, etc. when M9016 is on.	●	●	●	●
M9017	Data memory clear flag	OFF: No processing ON: Output clear	●Clears the unlatched data memory (other than special relays and special registers) in remote run mode from computer, etc. when M9017 is on.	●	●	●	●
M9020	User timing clock No. 0		●Relay which repeats on/off at intervals of predetermined scan. ●When power is turned on or reset is performed, the clock starts with off. ●Set the intervals of on/off by [DUTY] instruction. 	●	●	●	●
M9021	User timing clock No. 1						
M9022	User timing clock No. 2						
M9023	User timing clock No. 3						
M9024	User timing clock No. 4						
*2 M9025	Clock data set request	OFF: Ignore ON: Set requested	●Writes clock data from D9025-D9028 to the clock element after the [END] instruction is executed during the scan in which M9025 has changed from off to on.	●	●	●	–

Table 2.2 Special Relay List (Continue)

Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
M9026	Clock data error	OFF: No error ON: Error	● Switched on by clock data (D9025 to D9028) error.	●	●	●	—
M9027	Clock data display	OFF: Ignore ON: Display	● Clock data is read from D9025-D9028 and month, day, hour, minute and minute are indicated on the CPU front LED display.	—	—	●	—
*2 M9028	Clock data read request	OFF: Ignore ON: Read request	● Reads clock data to D9025-D9028 in BCD when M9028 is on.	●	●	●	—
M9030	0.1 second clock	0.05 seconds / 0.05 seconds	● 0.1 second, 0.2 second, 1 second, 2 second, and 1 minute clocks are generated. ● Not turned on and off per scan but turned on and off even during scan if corresponding time has elapsed. ● Starts with off when power is turned on or reset is performed.	●	●	●	●
M9031	0.2 second clock	0.1 seconds / 0.1 seconds					
M9032	1 second clock	0.5 seconds / 0.5 seconds					
M9033	2 second clock	1 seconds / 1 seconds					
M9034	1 minute clock	30 seconds / 30 seconds					
M9036	Normally ON	ON OFF	● Used as dummy contacts of initialization and application instruction in sequence program. ● M9036 and M9037 are turned on and off without regard to position of key switch on CPU front. M9038 and M9039 are under the same condition as RUN status except when the key switch is at STOP position, and turned off and on. Switched off if the key switch is in STOP position. M9038 is on for one scan only and M9039 is off for one scan only if the key switch is not in STOP position.	●	●	●	●
M9037	Normally OFF	ON OFF					
M9038	On only for 1 scan after run	ON / OFF, 1 scan					
M9039	RUN flag (off only for 1 scan after run)	ON / OFF, 1 scan					
M9040	PAUSE enable coil	OFF: PAUSE disabled ON: PAUSE enabled	● When RUN key switch is at PAUSE position or remote pause contact has turned on and if M9040 is on, PAUSE mode is set and M9041 is turned on.	●	●	●	●
M9041	PAUSE status contact	OFF: During pause ON: Not during pause					
M9042	Stop status contact	OFF: During stop ON: Not during stop	● Switched on when the RUN key switch is in STOP position.	●	●	●	●
M9043	Sampling trace completion	OFF: During sampling trace ON: Sampling trace completion	● Turned on upon completion of sampling trace performed the number of times preset by parameter after [STRA] instruction is executed. Reset when [STRAR] instruction is executed.	—	●	●	●
M9046	Sampling trace	OFF: Except during trace ON: During trace	● Switched on during sampling trace.	—	●	●	●
M9047	Sampling trace preparation	OFF: Sampling trace stop ON: Sampling trace start	● Switched on to start sampling trace. ● Switched off to stop sampling trace.	—	●	●	●
*2 M9050	Operation result storage memory change contact (for [CHG] instruction)	OFF: Not changed ON: Changed	● Switched on to exchange the operation result storage memory data and the save area data.	—	—	—	—
M9051	[CHG] instruction execution disable	OFF: Disable ON: Enable	● Switched on to disable the [CHG] instruction. ● Switched on when program transfer is requested and automatically switched off when transfer is complete.	—	—	●	●

Table 2.2 Special Relay List (Continue)

Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
*2 M9052	[SEG] instruction switching	OFF: 7SEG display ON: I/O partial refresh	● Switched on to execute the [SEG] instruction as an I/O partial refresh instruction. Switched off to execute the [SEG] instruction as a 7SEG display instruction.	●	●	●	●
*2 M9053	[EI] / [DI] instruction switching	OFF: Sequence interrupt control ON: Link interrupt control	● Switched on to execute the link refresh enable, disable (EI, DI) instructions.	●	●	●	—
M9054	STEP RUN flag	OFF: Other than step run ON: During step run	● Switched on when the RUN key switch is in STEP RUN position.	●	●	●	●
M9055	Status latch complete flag	OFF: Not complete ON: Complete	● Turned on when status latch is completed. Turned off by reset instruction.	—	●	●	●
M9056	Main program P, I set request	OFF: Other than P, I set request ON: P, I set request	● Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	—	—	●	●
M9057	Subprogram P, I set request	OFF: Except during P, I set request ON: During P, I set request					
*2 M9084	Error check	OFF: Checks enabled ON: Checks disabled	● Specify whether the following errors are to be checked or not after the [END] instruction is executed (to reduce END processing time): • Fuse blown • I/O unit verify error • Battery error	●	●	●	●
*2 *3 M9094	I/O change flag	OFF: Changed ON: Not changed	● After the head address of the required I/O module is set to D9094, switching M9094 on allows the I/O module to be changed in online mode. (One module is only allowed to be changed by one setting.) ● To be switched on in the program or peripheral device test mode to change the module during CPU RUN. To be switched on in peripheral device test mode to change the module during CPU STOP. ● RUN/STOP mode must not be changed until I/O module change is complete.	●	●	●	—

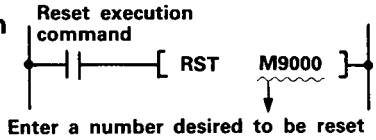
Table 2.2 Special Relay List

POINT

(1) Special relays are switched off when the PC is switched off or the RESET switch is set to LATCH CLEAR or RESET. Data remains unchanged when the RUN key switch is set to STOP.

(2) The above the relays with numbers marked *1 remain "on" if normal status is restored. Therefore, to turn them "off", use the following method:

1) Method by use program Insert the circuit shown at right into the program and turn on the reset execution command contact to clear the special relay M.



2) Perform forced reset by use of the test function of peripheral unit A6GPP, A6PHP, A6HGP. For the operation procedure, refer to the Instruction Manual for A6GPP, A6PHP, A6HGP.

3) By moving the RESET key switch at the CPU front to the RESET position, the special register is set to "0".

(3) Special relays marked *2 above are switched on/off in the sequence program.

(4) Special relays marked *3 above are switched on/off in test mode of the peripheral equipment.

2.18 Special Relays for Link

The link special relays are internal relays which are switched on/off by various factors occurring during data link operation. Their ON/OFF status will change if an error occurs during normal operation.
These relays are applicable to any CPUs.

1) Link special relays only valid when the host is the master station

Number	Name	Description	Details
M9200	[LRDP] instruction received	OFF: Unreceived ON: Received	● Depends on whether or not the [LRDP] (word device read) instruction has been received. ● Used in the program as an interlock for the [LRDP] instruction. ● Use the [RST] instruction to reset.
M9201	[LRDP] instruction complete	OFF: Incomplete ON: Complete	● Depends on whether or not the [LRDP] (word device read) instruction execution is complete. ● Used as a condition contact for resetting M9200 and M9201 after the [LRDP] instruction is complete. ● Use the [RST] instruction to reset.
M9202	[LWTP] instruction received	OFF: Unreceived ON: Received	● Depends on whether or not the [LWTP] (word device write) instruction has been received. ● Used in the program as an interlock for the [LWTP] instruction. ● Use the [RST] instruction to reset.
M9203	[LWTP] instruction complete	OFF: Incomplete ON: Complete	● Depends on whether or not the [LWTP] (word device write) instruction execution is complete. ● Used as a condition contact to reset M9202 and M9203 after the [LWTP] instruction is complete. ● Use the [RST] instruction to reset.
M9206	Link parameter error in the host	OFF: Normal ON: Error	Depends on whether or not the link parameter setting of the host is valid.
M9207	Link parameter unmatched between master stations	OFF: Normal ON: Unmatched	Depends on whether or not the link parameter setting of the master station in tier two matches that of the master station in tier three in a three-tier system. (Valid only for the master stations in a three-tier system.)
M9210	Link card error (master station)	OFF: Normal ON: Error	Depends on presence or absence of the link card hardware error. Judged by the CPU.
M9224	Link status	OFF: Offline ON: Online, station-to-station test, or self-loopback test	Depends on whether the master station is online or offline or is in station-to-station test or self-loopback test mode.

Table 2.3 Link Special Relay List (Continue)

Number	Name	Description	Details
M9225	Forward loop error	OFF: Normal ON: Error	Depends on the error condition of the forward loop line.
M9226	Reverse loop error	OFF: Normal ON: Error	Depends on the error condition of the reverse loop line.
M9227	Loop test status	OFF: Unexecuted ON: Forward or reverse loop test being executed	Depends on whether or not the master station is executing a forward or a reverse loop test.
M9232	Local station operating status	OFF: RUN or STEP RUN mode ON: STOP or PAUSE mode	Depends on whether or not a local station is in STOP or PAUSE mode.
M9233	Local station error detect	OFF: No error ON: Error detected	Depends on whether or not a local station has detected an error in another station.
M9235	Local or remote I/O station parameter error detect	OFF: No error ON: Error detected	Depends on whether or not a local or a remote I/O station has detected any link parameter error in the master station.
M9236	Local or remote I/O station initial communicating status	OFF: Noncommunicating ON: Communicating	Depends on whether or not a local or a remote I/O station is communicating initial data (such as parameters) with the master station.
M9237	Local or remote I/O station error	OFF: Normal ON: Error	Depends on the error condition of a local or remote I/O station.
M9238	Local or remote I/O station forward/ reverse loop error	OFF: Normal ON: Error	Depends on the error condition of the forward and reverse loop lines of a local or a remote I/O station.

Table 2.3 Link Special Relay List

2) Link special relays only valid when the host is a local station

Number	Name	Description	Details
M9204	[LRDP] instruction complete	OFF: Incomplete ON: Complete	On indicates that the [LRDP] instruction is complete at the local station.
M9205	[LWTP] instruction complete	OFF: Incomplete ON: Completed	On indicates that the [LWTP] instruction is complete at the local station.
M9211	Link card error (local station)	OFF: Normal ON: Error	Depends on presence or absence of the link card error. Judged by the CPU.
M9240	Link status	OFF: Online ON: Offline, station-to-station test, or self-loopback test	Depends on whether the local station is online or offline, or is in station-to-station test or self-loopback test mode.
M9241	Forward loop error	OFF: Normal ON: Error	Depends on the error condition of the forward loop line.
M9242	Reverse loop error	OFF: Normal ON: Error	Depends on the error condition of the reverse loop line.
M9243	Loopback execution	OFF: Non-executed ON: Executed	Depends on whether or not loopback is occurring at the local station.
M9246	Data unreceived	OFF: Received ON: Unreceived	Depends on whether or not data has been received from the master station.
M9247	Data unreceived	OFF: Received ON: Unreceived	Depends on on whether or not a tier three station has received data from its master station in a three-tier system.
M9250	Parameter unreceived	OFF: Received ON: Unreceived	Depends on whether or not link parameters have been received from the master station.
M9251	Link break	OFF: Normal ON: Break	Depends on the data link condition at the local station.
M9252	Loop test status	OFF: Unexecuted ON: Forward or reverse loop test is being executed.	Depends on whether or not the local station is executing a forward or a reverse loop test.
M9253	Master station operating status	OFF: RUN or STEP RUN mode ON: STOP or PAUSE mode	Depends on whether or not the master station is in STOP or PAUSE mode.
M9254	Operating status of other local stations	OFF: RUN or STEP RUN mode ON: STOP or PAUSE mode	Depends on whether or not a local station other than the host is in STOP or PAUSE mode.
M9255	Error status of other local stations	OFF: Normal ON: Error	Depends on whether or not a local station other than the host is in error.

Table 2.4 Link Special Relay List

2.19 Special Registers

Special registers are data registers of which applications have been determined inside the PC. Therefore, do not write data to the special registers in the program (except the ones with numbers marked *2 in the table).

Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
D9000	Fuse blow	Fuse blow module number	●When fuse blown modules are detected, the lowest number of detected units is stored in hexadecimal. (Example: When fuses of Y50 to 6F output modules have blown, "50" is stored in hexadecimal) To monitor the number by GPP or PU, perform monitor operation given in hexadecimal. (Cleared when all contents of D9100 to D9107 are reset to 0.)	●	●	●	●
D9002	I/O module verify error	I/O module verify error unit number	●If I/O modules, of which data are different from data entered, are detected when the power is turned on, the first I/O number of the lowest number unit among the detected units is stored in hexadecimal. (Storing method is the same as that of D9000.) To monitor the number by GPP or PU, perform monitor operation given in hexadecimal. (Cleared when all contents of D9116 to D9123 are reset to 0.)	●	●	●	●
*1 D9005	AC DOWN counter	AC DOWN count	●1 is added each time input voltage becomes 80% or less of rating while the CPU unit is performing operation, and the value is stored in BIN code.	●	●	●	●
*1 D9008	Self-diagnostic error	Self-diagnostic error number	●When error is found as a result of self-diagnosis, error number is stored in BIN code.	●	●	●	●
D9009	Annunciator detection	F number at which external failure has occurred	●When one of F0 to 255 is turned on by [OUT F] or [SET F], the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code. ●D9009 can be cleared by [RST F] or [LEDR] instruction. If another F number has been detected, the clearing of D9009 causes the next number to be stored in D9009.	●	●	—	—
			●When one of F0 to 255 is turned on by [OUT F] or [SET F], the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code. ●D9009 can be cleared by executing [RST F] or [LEDR] instruction or moving INDICATOR RESET switch on CPU front to ON position. If another F number has been detected, the clearing of D9009 causes the next number to be stored in D9009.	—	—	●	●
D9010	Error step	Step number at which operation error has occurred	●When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Thereafter, each time operation error occurs, the contents of D9010 are renewed.	●	●	●	—
D9011	Error step	Step number at which operation error has occurred	●When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Since storage into D9011 is made when M9011 changes from off to on, the contents of D9010 cannot be renewed unless M9011 is cleared by user program.	●	●	●	●
D9014	I/O control mode	I/O control mode number	●The I/O control mode set is returned in any of the following numbers: 0. Both input and output in direct mode 1. Input in refresh mode, output in direct mode 3. Both input and output in refresh mode	●	●	●	—

Table 2.5 Special Register List (Continue)

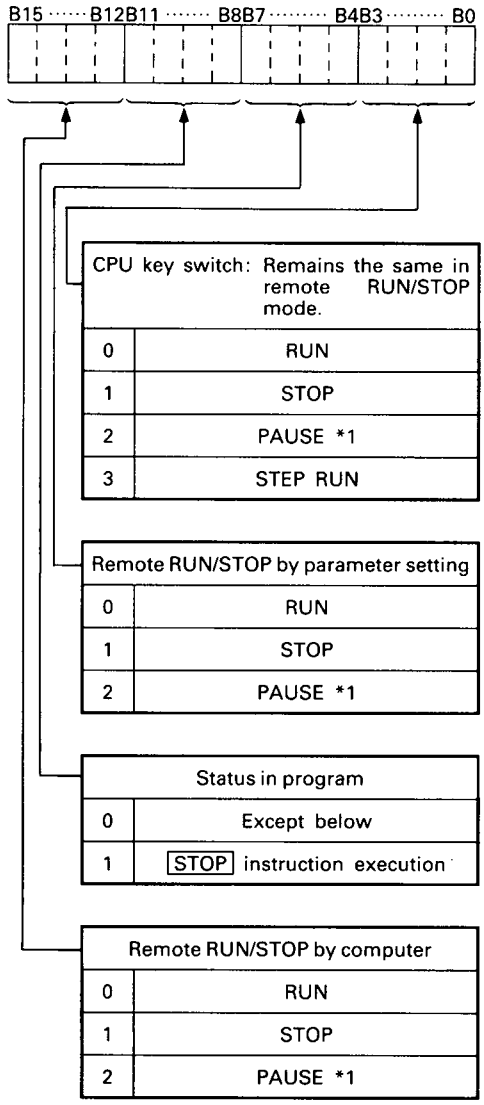
Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
D9015	CPU operating states	Operating states of CPU	● The operating states of CPU as shown below are stored in D9015.  CPU key switch: Remains the same in remote RUN/STOP mode. 0 RUN 1 STOP 2 PAUSE *1 3 STEP RUN Remote RUN/STOP by parameter setting 0 RUN 1 STOP 2 PAUSE *1 Status in program 0 Except below 1 [STOP] instruction execution Remote RUN/STOP by computer 0 RUN 1 STOP 2 PAUSE *1 *1: When the CPU is in RUN mode and M9040 is off, the CPU remains in RUN mode if changed to PAUSE mode.	●	●	●	●

Table 2.5 Special Register List (Continue)

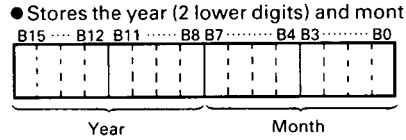
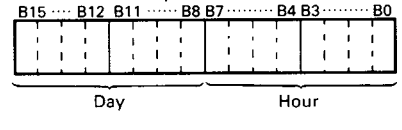
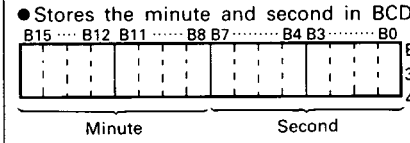
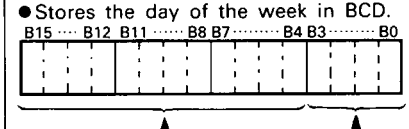
Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
D9016	ROM/RAM setting	0: ROM 1: RAM 2: E²ROM	● Indicates the setting of memory select chip. One value of 0 to 2 is stored in BIN code.	●	—	—	—
	Program number	0: Main program (ROM) 1: Main program (RAM) 2: Subprogram (RAM)	● Indicates which sequence program is run presently. One value of 0 to 2 is stored in BIN code. (2 is only valid for the A3N and A3HCPU.)	—	●	●	●
D9017	Scan time	Minimum scan time (per 10ms)	● If scan time is smaller than the content of D9017, the value is newly stored at each END. Namely, the minimum value of scan time is stored into D9017 in BIN code.	●	●	●	●
D9018	Scan time	Scan time (per 10ms)	● Scan time is stored in BIN code at each END and always rewritten.	●	●	●	●
D9019	Scan time	Maximum scan time (per 10ms)	● If scan time is larger than the content of D9019, the value is newly stored at each END. Namely, the maximum value of scan time is stored into D9019 in BIN code.	●	●	●	●
*2 D9020	Constant scan	Constant scan time (Set by user in 10ms increments)	● Sets the interval between consecutive program starts in multiples of 10ms. 0: No setting 1 to 200: Set. Program is executed at intervals of (set value) × 10ms.	●	●	●	●
*2 D9025	Clock data	(Year, month)	● Stores the year (2 lower digits) and month in BCD.  Example: 1987, July H8707	●	●	●	—
*2 D9026	Clock data	(Day, hour)	● Stores the day and hour in BCD.  Example: 31th, 10 o'clock H3110	●	●	●	—
*2 D9027	Clock data	(Minute, second)	● Stores the minute and second in BCD.  Example: 35 minutes, 48 seconds H3548	●	●	●	—
*2 D9028	Clock data	(, day of the week)	● Stores the day of the week in BCD.  Example: Friday H0005 0 must be set. Day of the week: 0 Sunday; 1 Monday; 2 Tuesday; 3 Wednesday; 4 Thursday; 5 Friday; 6 Saturday	●	●	●	—

Table 2.5 Special Register List (Continue)

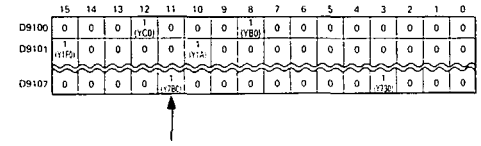
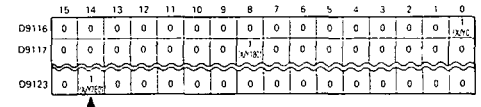
Number	Name	Description	Details	Applicable CPU			
				A1N	A2N	A3N	A3H
D9044	For sampling trace	Step or time for sampling trace	●The BIN value in D9044 is used as a sampling trace condition when M9044 is switched on/off by the peripheral device to trigger the sampling trace. Scan ———— 0 Time interval — time (in 10ms increments)				
*2 *3 D9094	Changed I/O module head address	Changed I/O module head address	●Stores the most significant digits of the head address of the I/O module changed in online mode. Example: Input module X2F0 H2F	●	●	●	—
*1 D9100 *1 D9101 *1 D9102 *1 D9103 *1 D9104 *1 D9105 *1 D9106 *1 D9107	Fuse blown module	Bit pattern in units of 16 points of fuse blow modules	●Output module numbers (in units of 16 points), of which fuses have blown, are entered in bit pattern. (Preset output unit numbers when parameter setting has been performed.)  Indicates fuse blow. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)	●	●	●	●
*1 D9116 *1 D9117 *1 D9118 *1 D9119 *1 D9120 *1 D9121 *1 D9122 *1 D9123	I/O module verify error	Bit pattern in units of 16 points of verify error units	●When I/O modules, of which data are different from those entered at power-on, have been detected, the I/O unit numbers (in units of 16 points) are entered in bit pattern. (Preset I/O unit numbers when parameter setting has been performed.)  Indicates I/O module verify error. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)	●	●	●	●

Table 2.5 Special Register List (Continue)

<table>
<tr><th rowspan="2">Number</th><th rowspan="2">Name</th><th rowspan="2">Description</th><th rowspan="2">Details</th><th colspan="4">Applicable CPU</th></tr>
<tr><th>A1N</th><th>A2N</th><th>A3N</th><th>A3H</th></tr>
<tr><td>D9124</td><td>Annunciator detection quantity</td><td>Annunciator detection quantity</td><td>● When one of F0 to 255 is turned on by [OUT F] or [SET F] 1 is added to the contents of D9124. When [RST F] or [LEDR] instruction is executed, 1 is subtracted from the contents of D9124. (For A3(E)CPU, it can be performed by use of INDICATOR RESET switch on front face of CPU unit.)
● Quantity, which has been turned on by [OUT F] or [SET F] is stored into D9124 in BIN code. The value of D9124 is maximum 8.</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>D9125
D9126
D9127
D9128
D9129
D9130
D9131
D9132</td><td>Annunciator detection number</td><td>Annunciator detection number</td><td>● When one of F0 to 255 is turned on by [OUT F] or [SET F], F number, which has turned on, is entered into D9125 to D9132 in due order in BIN code.
● F number, which has been turned off by [RST F], is erased from D9125 to D9132, and the contents of data registers succeeding the data register, where the erased F number was stored, are shifted to the preceding data registers.
● By executing [LEDR] instruction, the contents of D9125 to D9132 are shifted upward by one. (For A3N, A3HCPU, it can be performed by use of INDICATOR RESET switch on front of CPU module.)
● When there are 8 annunciator detections, the 9th one is not stored into D9125 to 9132 even if detected.

SET SET SET RST SET SET SET SET SET SET SET
F50 F25 F19 F25 F15 F70 F65 F38 F110 F151 F210 LEDR
D9009 0 50 50 50 50 50 50 50 50 50 50 50 99
D9124 0 1 2 3 2 3 4 5 6 7 8 8 8
D9125 0 50 50 50 50 50 50 50 50 50 50 50 99
D9126 0 0 25 25 99 99 99 99 99 99 99 99 15
D9127 0 0 0 99 0 15 15 15 15 15 15 15 70
D9128 0 0 0 0 0 0 70 70 70 70 70 70 65
D9129 0 0 0 0 0 0 65 65 65 65 65 65 38
D9130 0 0 0 0 0 0 0 38 38 38 38 38 110
D9131 0 0 0 0 0 0 0 0 0 110 110 110 151
D9132 0 0 0 0 0 0 0 0 0 0 151 151 210</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
</table>

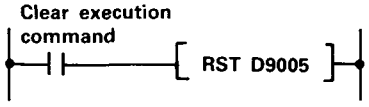
Table 2.5 Special Register List

POINT

(1) Special registers are cleared when the PC is switched off or the RESET switch is set to LATCH CLEAR or RESET. Data remains unchanged when the RUN key switch is set to STOP.

(2) Special registers marked *1 above are latched and their data will remain unchanged after normal status is restored.

1) Method by user program
Insert the circuit shown at right into the program and turn on the clear execution command contact to clear the contents of register.



2) Method by peripheral equipment (A6GPP, A6PHP, A6HGP, A7PU)
Set the register to "0" by changing the present value by the test function of peripheral equipment (A6GPP, A6PHP, A6HGP, A7PU) or set to "0" by forced reset. For the operation procedure, refer to the Instruction Manual for A6GPP, A6PHP, A6HGP.

3) By moving the RESET key switch at the CPU front to the RESET position, the special register is set to "0".

(3) Data is written to special registers marked *2 above in the sequence program.

(4) Data is written to special registers marked *3 above in test mode of the peripheral device.

2.20 Special Registers for Link

The link special register stores the result of any error, etc. which may occur during data communication as a numeric value.
By monitoring the link special register, any station number with an error or fault diagnosis can be read.
These registers are applicable to any CPUs.

1) Link special registers only valid when the host station is the master station

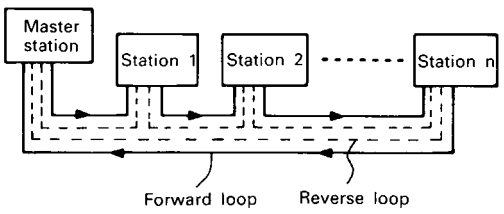
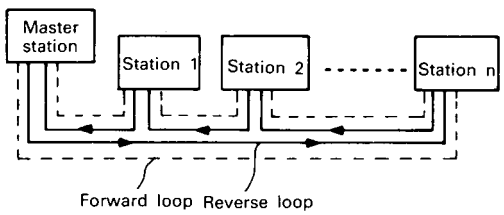
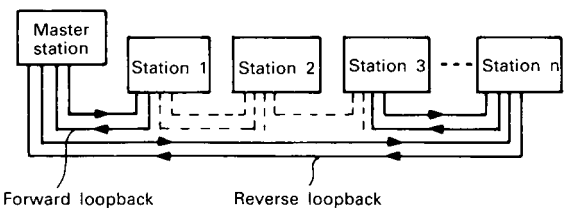
Number	Name	Description	Details
D9200	[LRDP] processing result	0: Normal 2: [LRDP] instruction setting fault 3: Corresponding station error 4: [LRDP] cannot be executed in the corresponding station	Stores the execution result of the [LRDP] (word device read) instruction. ● [LRDP] instruction setting fault: Faulty setting of the [LRDP] instruction constant, source, and/or destination. ● Corresponding station error: One of the stations is not communicating. ● [LRDP] cannot be executed in the corresponding station: The specified station is a remote I/O station.
D9201	[LWTP] processing result	0: Normal 2: [LWTP] instruction setting fault 3: Corresponding station error 4: [LWTP] cannot be executed in the corresponding station	Stores the execution result of the [LWTP] (word device write) instruction. ● [LWTP] instruction setting fault: Faulty setting of the [LWTP] instruction constant, source, and/or destination. ● Corresponding station error: One of the stations is not communicating. ● [LWTP] cannot be executed in the corresponding station: The specified station is a remote I/O station.
D9204 (Continue)	Link status	0: Data link in forward loop 1: Data link in reverse loop 2: Loopback in forward/reverse direction 3: Loopback in forward direction 4: Loopback in reverse direction 5: Data link impossible	Stores the present path status of the data link. ● Data link in forward loop  ● Data link in reverse loop  ● Loopback in forward/reverse loops 

Table 2.6 Link Special Register List (Continue)

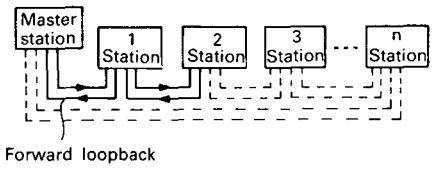
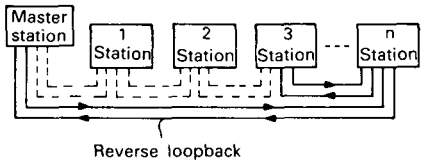
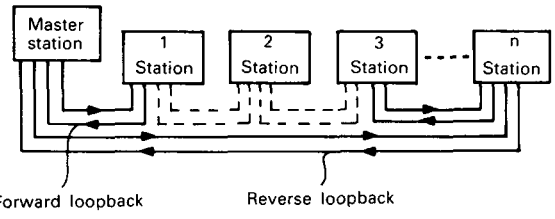
Number	Name	Description	Details
D9204	Link status		● Loopback in forward loop only  ● Loopback in reverse loop only 
D9205	Loopback executing station	Station executing forward loopback	Stores the local or remote I/O station number at which loopback is being executed.  In the above example, 1 is stored into D9205 and 3 into D9206. If data link returns to normal status (data link in forward loop), values in D9205 and D9206 remain 1 and 3. Reset using sequence program or the RESET key.
D9206	Loopback executing station	Station executing reverse loopback	
D9207	Link scan time	Maximum value	Stores the data link processing time with all local and remote I/O stations. ● Input (X), output (Y), link relay (B), and link register (W) assigned in link parameters communicate with the corresponding stations every link scan. ● Link scan is a period of time during which data link is executed with all connected slave stations, independently of the sequence program scan time.
D9208	Link scan time	Minimum value	
D9209	Link scan time	Present value	
D9210	Retry count	Total number stored	Stores the number of retry times due to transmission error. Count stops at a maximum of "FFFF$_H$". RESET to return the count to 0.
D9211	Loop switching count	Total number stored	Stores the number of times the loop line has been switched to reverse loop or loopback.

Table 2.6 Link Special Register List (Continue)

<table>
<tr><th>Number</th><th>Name</th><th>Description</th><th>Details</th></tr>
<tr><td>D9212</td><td>Local station operating status</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local station numbers which are in STOP or PAUSE mode.
<table><tr><th rowspan="2">Device Number</th><th colspan="16">Bit</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9212</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>D9213</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>D9214</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>D9215</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table>When a local station is switched to STOP or PAUSE mode, the bit corresponding to the station number in the register becomes "1".
Example: When station 7 switches to STOP mode, bit 6 in D9212 becomes "1", and when D9212 is monitored, its value is "64 (40H)".</td></tr>
<tr><td>D9213</td><td>Local station operating status</td><td>Stores the status of stations 17 to 32</td></tr>
<tr><td>D9214</td><td>Local station operating status</td><td>Stores the status of stations 33 to 48</td></tr>
<tr><td>D9215</td><td>Local station operating status</td><td>Stores the status of stations 49 to 64</td></tr>
<tr><td>D9216</td><td>Local station error detection</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local station numbers which are in error.
<table><tr><th rowspan="2">Device Number</th><th colspan="16">Bit</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9216</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>D9217</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>D9218</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>D9219</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table>If a local station detects an error, the bit corresponding to the station number becomes "1".
Example: When station 6 and 12 detect an error, bits 5 and 11 in D9216 become "1", and when D9216 is monitored, its value is "2080 (820H)".</td></tr>
<tr><td>D9217</td><td>Local station error detection</td><td>Stores the status of stations 17 to 32</td></tr>
<tr><td>D9218</td><td>Local station error detection</td><td>Stores the status of stations 33 to 48</td></tr>
<tr><td>D9219</td><td>Local station error detection</td><td>Stores the status of stations 49 to 64</td></tr>
<tr><td>D9220</td><td>Local station parameter mismatched or remote station I/O assignment error</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local station numbers which contain mismatched parameters or of remote station numbers for which incorrect I/O assignment has been made.
<table><tr><th rowspan="2">Device Number</th><th colspan="16">Bit</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9220</td><td>L/R 16</td><td>L/R 15</td><td>L/R 14</td><td>L/R 13</td><td>L/R 12</td><td>L/R 11</td><td>L/R 10</td><td>L/R 9</td><td>L/R 8</td><td>L/R 7</td><td>L/R 6</td><td>L/R 5</td><td>L/R 4</td><td>L/R 3</td><td>L/R 2</td><td>L/R 1</td></tr><tr><td>D9221</td><td>L/R 32</td><td>L/R 31</td><td>L/R 30</td><td>L/R 29</td><td>L/R 28</td><td>L/R 27</td><td>L/R 26</td><td>L/R 25</td><td>L/R 24</td><td>L/R 23</td><td>L/R 22</td><td>L/R 21</td><td>L/R 20</td><td>L/R 19</td><td>L/R 18</td><td>L/R 17</td></tr><tr><td>D9222</td><td>L/R 48</td><td>L/R 47</td><td>L/R 46</td><td>L/R 45</td><td>L/R 44</td><td>L/R 43</td><td>L/R 42</td><td>L/R 41</td><td>L/R 40</td><td>L/R 39</td><td>L/R 38</td><td>L/R 37</td><td>L/R 36</td><td>L/R 35</td><td>L/R 34</td><td>L/R 33</td></tr><tr><td>D9223</td><td>L/R 64</td><td>L/R 63</td><td>L/R 62</td><td>L/R 61</td><td>L/R 60</td><td>L/R 59</td><td>L/R 58</td><td>L/R 57</td><td>L/R 56</td><td>L/R 55</td><td>L/R 54</td><td>L/R 53</td><td>L/R 52</td><td>L/R 51</td><td>L/R 50</td><td>L/R 49</td></tr></table>If a local station acting as the master station of tier three detects a parameter error or a remote station contains an invalid I/O assignment, the bit corresponding to the station number becomes "1".
Example: When local station 5 and remote I/O station 14 detect an error, bits 4 and 13 in D9220 become "1", and when D9220 is monitored, its value is "8208 (2010H)".</td></tr>
<tr><td>D9221</td><td>Local station parameter mismatched or remote station I/O assignment error</td><td>Stores the status of stations 17 to 32.</td></tr>
<tr><td>D9222</td><td>Local station parameter mismatched or remote station I/O assignment error</td><td>Stores the status of stations 33 to 48.</td></tr>
<tr><td>D9223</td><td>Local station parameter mismatched or remote station I/O assignment error</td><td>Stores the status of stations 49 to 64.</td></tr>
<tr><td>D9224</td><td>Initial communication between local or remote I/O stations</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local or remote station numbers while they are communicating the initial data with their relevant master station.
<table><tr><th rowspan="2">Device Number</th><th colspan="16">Bit</th></tr><tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9224</td><td>L/R 16</td><td>L/R 15</td><td>L/R 14</td><td>L/R 13</td><td>L/R 12</td><td>L/R 11</td><td>L/R 10</td><td>L/R 9</td><td>L/R 8</td><td>L/R 7</td><td>L/R 6</td><td>L/R 5</td><td>L/R 4</td><td>L/R 3</td><td>L/R 2</td><td>L/R 1</td></tr><tr><td>D9225</td><td>L/R 32</td><td>L/R 31</td><td>L/R 30</td><td>L/R 29</td><td>L/R 28</td><td>L/R 27</td><td>L/R 26</td><td>L/R 25</td><td>L/R 24</td><td>L/R 23</td><td>L/R 22</td><td>L/R 21</td><td>L/R 20</td><td>L/R 19</td><td>L/R 18</td><td>L/R 17</td></tr><tr><td>D9226</td><td>L/R 48</td><td>L/R 47</td><td>L/R 46</td><td>L/R 45</td><td>L/R 44</td><td>L/R 43</td><td>L/R 42</td><td>L/R 41</td><td>L/R 40</td><td>L/R 39</td><td>L/R 38</td><td>L/R 37</td><td>L/R 36</td><td>L/R 35</td><td>L/R 34</td><td>L/R 33</td></tr><tr><td>D9227</td><td>L/R 64</td><td>L/R 63</td><td>L/R 62</td><td>L/R 61</td><td>L/R 60</td><td>L/R 59</td><td>L/R 58</td><td>L/R 57</td><td>L/R 56</td><td>L/R 55</td><td>L/R 54</td><td>L/R 53</td><td>L/R 52</td><td>L/R 51</td><td>L/R 50</td><td>L/R 49</td></tr></table>The bit corresponding to the station number which is currently communicating the initial settings becomes "1".
Example: When stations 23 and 45 are communicating, bit 6 of D9225 and bit 12 of D9226 become "1", and when D9225 is monitored, its value is "64 (40H)", and when D9226 is monitored, its value is "4096 (1000H)".</td></tr>
<tr><td>D9225</td><td>Initial communication between local or remote I/O stations</td><td>Stores the status of stations 17 to 32</td></tr>
<tr><td>D9226</td><td>Initial communication between local or remote I/O stations</td><td>Stores the status of stations 33 to 48</td></tr>
<tr><td>D9227</td><td>Initial communication between local or remote I/O stations</td><td>Stores the status of stations 49 to 64</td></tr>
</table>

Table 2.6 Link Special Register List (Continue)

<table>
<tr><th>Number</th><th>Name</th><th>Description</th><th>Details</th></tr>
<tr><td>D9228</td><td>Local or remote I/O station error</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local or remote station numbers which are in error.
<table><tr><th>Device Number</th><th colspan="16">Bit</th></tr><tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9228</td><td>L/R 16</td><td>L/R 15</td><td>L/R 14</td><td>L/R 13</td><td>L/R 12</td><td>L/R 11</td><td>L/R 10</td><td>L/R 9</td><td>L/R 8</td><td>L/R 7</td><td>L/R 6</td><td>L/R 5</td><td>L/R 4</td><td>L/R 3</td><td>L/R 2</td><td>L/R 1</td></tr><tr><td>D9229</td><td>L/R 32</td><td>L/R 31</td><td>L/R 30</td><td>L/R 29</td><td>L/R 28</td><td>L/R 27</td><td>L/R 26</td><td>L/R 25</td><td>L/R 24</td><td>L/R 23</td><td>L/R 22</td><td>L/R 21</td><td>L/R 20</td><td>L/R 19</td><td>L/R 18</td><td>L/R 17</td></tr><tr><td>D9230</td><td>L/R 48</td><td>L/R 47</td><td>L/R 46</td><td>L/R 45</td><td>L/R 44</td><td>L/R 43</td><td>L/R 42</td><td>L/R 41</td><td>L/R 40</td><td>L/R 39</td><td>L/R 38</td><td>L/R 37</td><td>L/R 36</td><td>L/R 35</td><td>L/R 34</td><td>L/R 33</td></tr><tr><td>D9231</td><td>L/R 64</td><td>L/R 63</td><td>L/R 62</td><td>L/R 61</td><td>L/R 60</td><td>L/R 59</td><td>L/R 58</td><td>L/R 57</td><td>L/R 56</td><td>L/R 55</td><td>L/R 54</td><td>L/R 53</td><td>L/R 52</td><td>L/R 51</td><td>L/R 50</td><td>L/R 49</td></tr></table>The bit corresponding to the station number with the error becomes "1".
Example: When local station 3 and remote I/O station 14 have an error, bits 2 and 13 of D9228 become "1", and when D9228 is monitored, its value is "8196 (2004H)".</td></tr>
<tr><td>D9229</td><td>Local or remote I/O station error</td><td>Stores the status of stations 17 to 32</td></tr>
<tr><td>D9230</td><td>Local or remote I/O station error</td><td>Stores the status of stations 33 to 48</td></tr>
<tr><td>D9231</td><td>Local or remote I/O station error</td><td>Stores the status of stations 49 to 64</td></tr>
<tr><td>D9232</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 1 to 8</td><td rowspan="8">Stores the local or remote station number at which a forward or reverse loop error has occurred.
<table><tr><th>Device Number</th><th colspan="16">Bit</th></tr><tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9232</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R8</td><td colspan="2">L/R7</td><td colspan="2">L/R6</td><td colspan="2">L/R5</td><td colspan="2">L/R4</td><td colspan="2">L/R3</td><td colspan="2">L/R2</td><td colspan="2">L/R1</td></tr><tr><td>D9233</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R16</td><td colspan="2">L/R15</td><td colspan="2">L/R14</td><td colspan="2">L/R13</td><td colspan="2">L/R12</td><td colspan="2">L/R11</td><td colspan="2">L/R10</td><td colspan="2">L/R9</td></tr><tr><td>D9234</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R24</td><td colspan="2">L/R23</td><td colspan="2">L/R22</td><td colspan="2">L/R21</td><td colspan="2">L/R20</td><td colspan="2">L/R19</td><td colspan="2">L/R18</td><td colspan="2">L/R17</td></tr><tr><td>D9235</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R32</td><td colspan="2">L/R31</td><td colspan="2">L/R30</td><td colspan="2">L/R29</td><td colspan="2">L/R28</td><td colspan="2">L/R27</td><td colspan="2">L/R26</td><td colspan="2">L/R25</td></tr><tr><td>D9236</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R40</td><td colspan="2">L/R39</td><td colspan="2">L/R38</td><td colspan="2">L/R37</td><td colspan="2">L/R36</td><td colspan="2">L/R35</td><td colspan="2">L/R34</td><td colspan="2">L/R33</td></tr><tr><td>D9237</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R48</td><td colspan="2">L/R47</td><td colspan="2">L/R46</td><td colspan="2">L/R45</td><td colspan="2">L/R44</td><td colspan="2">L/R43</td><td colspan="2">L/R42</td><td colspan="2">L/R41</td></tr><tr><td>D9238</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R56</td><td colspan="2">L/R55</td><td colspan="2">L/R54</td><td colspan="2">L/R53</td><td colspan="2">L/R52</td><td colspan="2">L/R51</td><td colspan="2">L/R50</td><td colspan="2">L/R49</td></tr><tr><td>D9239</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td></tr><tr><td></td><td colspan="2">L/R64</td><td colspan="2">L/R63</td><td colspan="2">L/R62</td><td colspan="2">L/R61</td><td colspan="2">L/R60</td><td colspan="2">L/R59</td><td colspan="2">L/R58</td><td colspan="2">L/R57</td></tr></table>In the above table, "F" indicates a forward loop line and "R" a reverse loop line. The bit corresponding to the station number at which the forward or reverse loop error has occurred, becomes "1".
Example: When the forward loop line of station 5 has an error, bit 8 of D9232 become "1", and when D9232 is monitored, its value is "256 (100H)".</td></tr>
<tr><td>D9233</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 9 to 16</td></tr>
<tr><td>D9234</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 17 to 24</td></tr>
<tr><td>D9235</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 25 to 32</td></tr>
<tr><td>D9236</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 33 to 40</td></tr>
<tr><td>D9237</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 41 to 48</td></tr>
<tr><td>D9238</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 49 to 56</td></tr>
<tr><td>D9239</td><td>Local or remote I/O station loop error</td><td>Stores the status of stations 57 to 64</td></tr>
<tr><td>D9240</td><td>Number of receive error detection times</td><td>Total number stored</td><td>Stores the number of times the following transmission errors have been detected:
CRC, OVER, AB.IF
Count is made to a maximum of FFFFH. RESET to return the count to 0.</td></tr>
</table>

Table 2.6 Link Special Register List

2) Link special registers only valid when the host station is a local station

<table>
<tr><th>Number</th><th>Name</th><th>Description</th><th>Details</th></tr>
<tr><td>D9243</td><td>Own station number check</td><td>Stores a station number. (0 to 64)</td><td>Allows a local station to confirm its own station number.</td></tr>
<tr><td>D9244</td><td>Total number of slave stations</td><td>Stores the number of slave stations</td><td>Indicates the number of slave stations in one loop.</td></tr>
<tr><td>D9245</td><td>Number of receive error detection times</td><td>Total number stored</td><td>Stores the number of times the following transmission errors have been detected:
CRC, OVER, AB.IF
Count is made to a maximum of FFFF_H. RESET to return the count to 0.</td></tr>
<tr><td>D9248</td><td>Local station operating status</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local station number which is in STOP or PAUSE mode.
<table><tr><th>Device Number</th><th colspan="16">Bit</th></tr><tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9248</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>D9249</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>D9250</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>D9251</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table>The bit corresponding to the station number which is in STOP or PAUSE mode, becomes "1".
Example: When local stations 7 and 15 are in STOP mode, bits 6 and 14 of D9248 become "1", and when D9248 is monitored, its value is "16448 (4040H)".</td></tr>
<tr><td>D9249</td><td>Local station operating status</td><td>Stores the status of stations 17 to 32</td></tr>
<tr><td>D9250</td><td>Local station operating status</td><td>Stores the status of stations 33 to 48</td></tr>
<tr><td>D9251</td><td>Local station operating status</td><td>Stores the status of stations 49 to 64</td></tr>
<tr><td>D9252</td><td>Local station error</td><td>Stores the status of stations 1 to 16</td><td rowspan="4">Stores the local station number other than the host, which is in error.
<table><tr><th>Device Number</th><th colspan="16">Bit</th></tr><tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr><tr><td>D9252</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr><tr><td>D9253</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr><tr><td>D9254</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr><tr><td>D9255</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr></table>The bit corresponding to the station number which is in error, becomes "1".
Example: When local station 12 is in error, bit 11 of D9252 becomes "1", and when D9252 is monitored, its value is "2048 (800H)".</td></tr>
<tr><td>D9253</td><td>Local station error</td><td>Stores the status of stations 17 to 32</td></tr>
<tr><td>D9254</td><td>Local station error</td><td>Stores the status of stations 33 to 48</td></tr>
<tr><td>D9255</td><td>Local station error</td><td>Stores the status of stations 49 to 64</td></tr>
</table>

Table 2.7 Link Special Register List

2.21 Assignment of I/O Addresses

I/O addresses must be assigned to I/O module devices before data communication between the CPU and I/O modules.

2.21.1 Basics of assignment

I/O assignment is optional.
I/O addresses must be calculated as follows according to whether I/O assignment has been made or not.

(1) No parameter I/O assignment

1) Assign I/O numbers in order of extension base stage setting numbers (as opposed to order of extension cable connection).
2) Assign I/O numbers to the main base and extension base(s) on the assumption that each base has 8 slots. (The final 3 slots on a 5 slot base must be accounted for and represent empty I/O slots.)
3) Assign 16 points to an empty slot.
4) Any extension stage which has been skipped must be accounted for as 8 vacant slots.

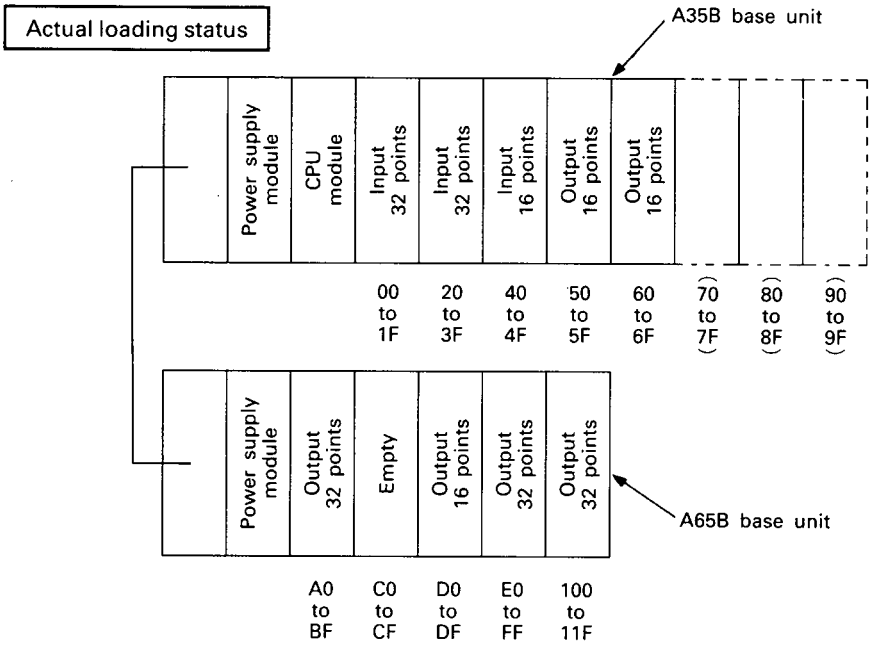
(2) "I/O ASSIGNMENT" has been made

1) 16, 32, 48 or 64 inputs, outputs, vacancies (i.e. a reservation for later expansion) or special function module I/O points can be assigned to each I/O slot.
2) A special function module will not operate correctly if it is located in a slot allocated to the wrong type of module (i.e. X or Y) or if the I/O count is wrong.
3) Any slot for which an I/O setting has not been made uses the actual I/O capacity of the module loaded.
4) The parameter I/O allocation takes priority over the actual module's I/O capacity.

   Hence: The second 16 I/O points of a 32 way module placed in a slot with 16 points allocated, will be unusable, and,
   The second 16 I/O points in a slot with 32 points allocated but containing a 16 way module will be unusable.
5) Any I/O module located in a slot allocated as vacant, will be unusable.
6) I/O assignment must be made to any remote I/O station in a data link system.
7) Any I/O combined module (e.g. A42XY) should be allocated as an output module.

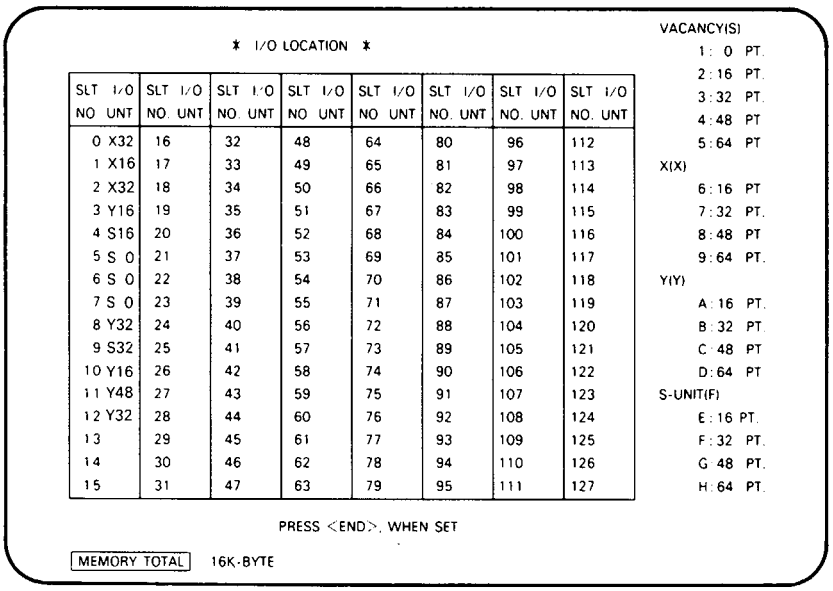
2.21.2 I/O address assignment examples

(1) No parameter I/O assignment



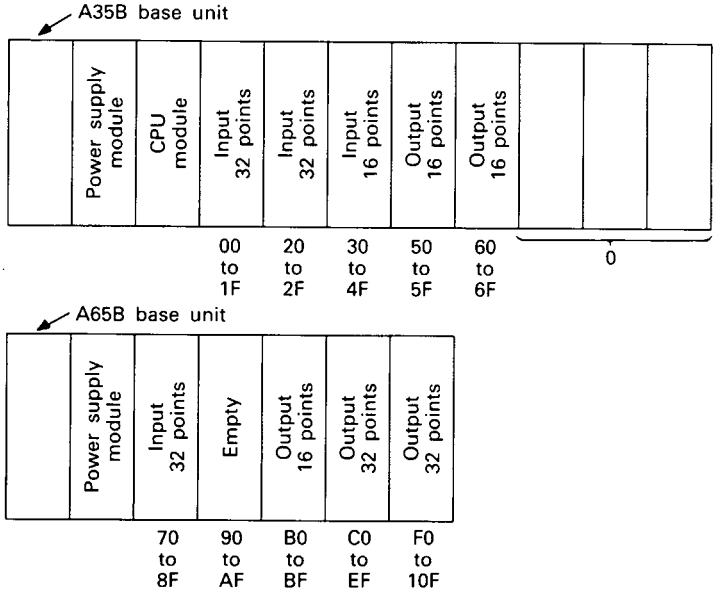
(2) "I/O ASSIGNMENT" has been made

1) I/O assignment



2) I/O addresses after making "I/O ASSIGNMENT"

Actual loading status



3. PROGRAMMING

3.1 Introduction

The ACPU may be programmed in any of the relay symbol, logic symbolic and MELSAP languages.
For the MELSAP language, see the MELSAP (SW□-SAPA) Programming Manual.
The relay symbol language is based on the concept of the relay control ladder. The PC uses a series processing system in which operation is performed in accordance with programs read sequentially.

(1) Program processing

The ladder in Fig. 3.1 is processed by the PC left to right and top to bottom in series. After the END instruction is executed, processing returns to step 0 and repeats operation repeatedly. PC ladders have advantages in that a sneak path prevention diode is not required and there is no limit to the number of auxiliary contacts used.

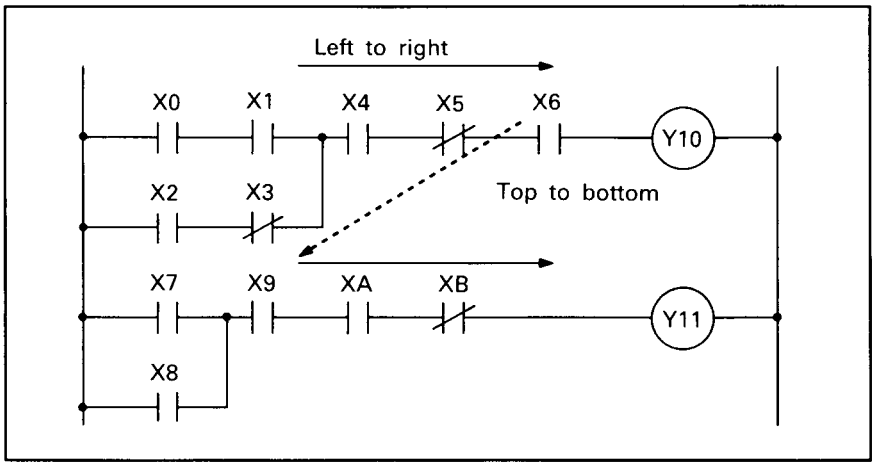


Fig. 3.1 Operation Processing Sequence

(2) Relay symbol, logic symbolic languages

The relay symbol language is based on relay symbol representations and allows any ladder to be programmed in the form very close to the relay control sequence ladder.
The logic symbolic language is based on the assembly language, one of the languages used to write microcomputer programs, and represents any program in the instruction, source and destination parts.
Any program written in either language is stored onto the memory after it is converted into the machine language.

IB (NA) 66147-A

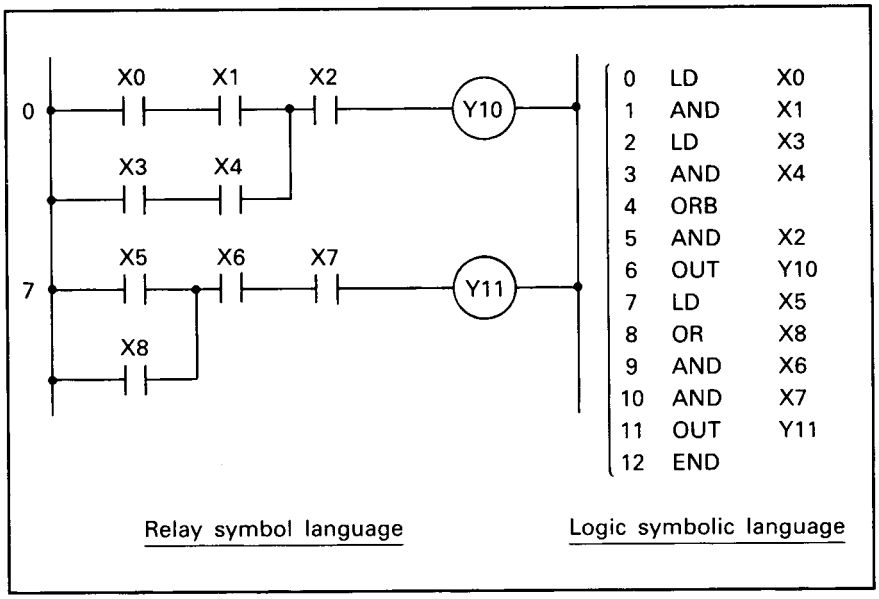


Fig. 3.2 Relay Symbol, Logic Symbolic Languages

3.2 Numeric Value and Character Representations

All numeric values and characters are processed in BIN (binary) by the PC CPU.
All data can be represented by two states, 0 and 1. The PC CPU has conversion functions between BIN and DEC (decimal) and between BIN and HEX (hexadecimal) so that programs can be written and operation results monitored in DEC or HEX. The ACPU can process 16- and 32-bit data.

(1) BIN

1) Indicates numeric values represented by 0 and 1.
A carry occurs after 1 in the BIN system.

BIN	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010
DEC	0	1	2	3	4	5	6	7	8	9	10

The shaded area indicates that a carry has occurred.

2) BIN bits correspond to DEC values as indicated below:

BIN bit	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
DEC value	1024	512	256	128	64	32	16	8	4	2	1

Example: $1100101 = 2^6 + 2^5 + 2^2 + 2^0 = 64 + 32 + 4 + 1 = 101$
BIN 1100101 corresponds to DEC 101.

(2) BCD (Binary Coded Decimal)

A code for representing DEC digits in a BIN format. A carry occurs after 9 in the BCD system.

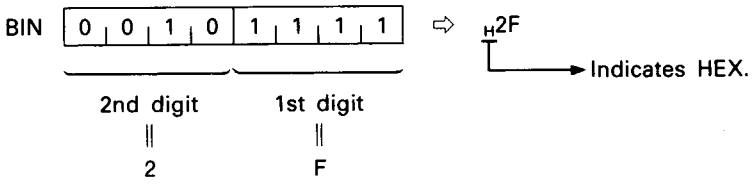
DEC	BIN	BCD
0	0	0
1	1	1
2	10	10
3	11	11
4	100	100
5	101	101
6	110	110
7	111	111
8	1000	1000
9	1001	1001
10	1010	1 0000
11	1011	1 0001
12	1100	1 0010

(3) HEX

1) 9 is followed by A, B, C, D, E, and F in the HEX system. A carry occurs after F.

DEC	HEX	BCD
0	0	0
1	1	1
2	2	10
3	3	11
⋮	⋮	⋮
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111
16	10	10000
17	11	10001
⋮	⋮	⋮
47	2F	101111

2) HEX corresponds to BIN as indicated below:



3

(4) ASCII

1) ASCII codes correspond to alphanumeric characters and special symbols and are used to communicate data between the PC and external equipment.

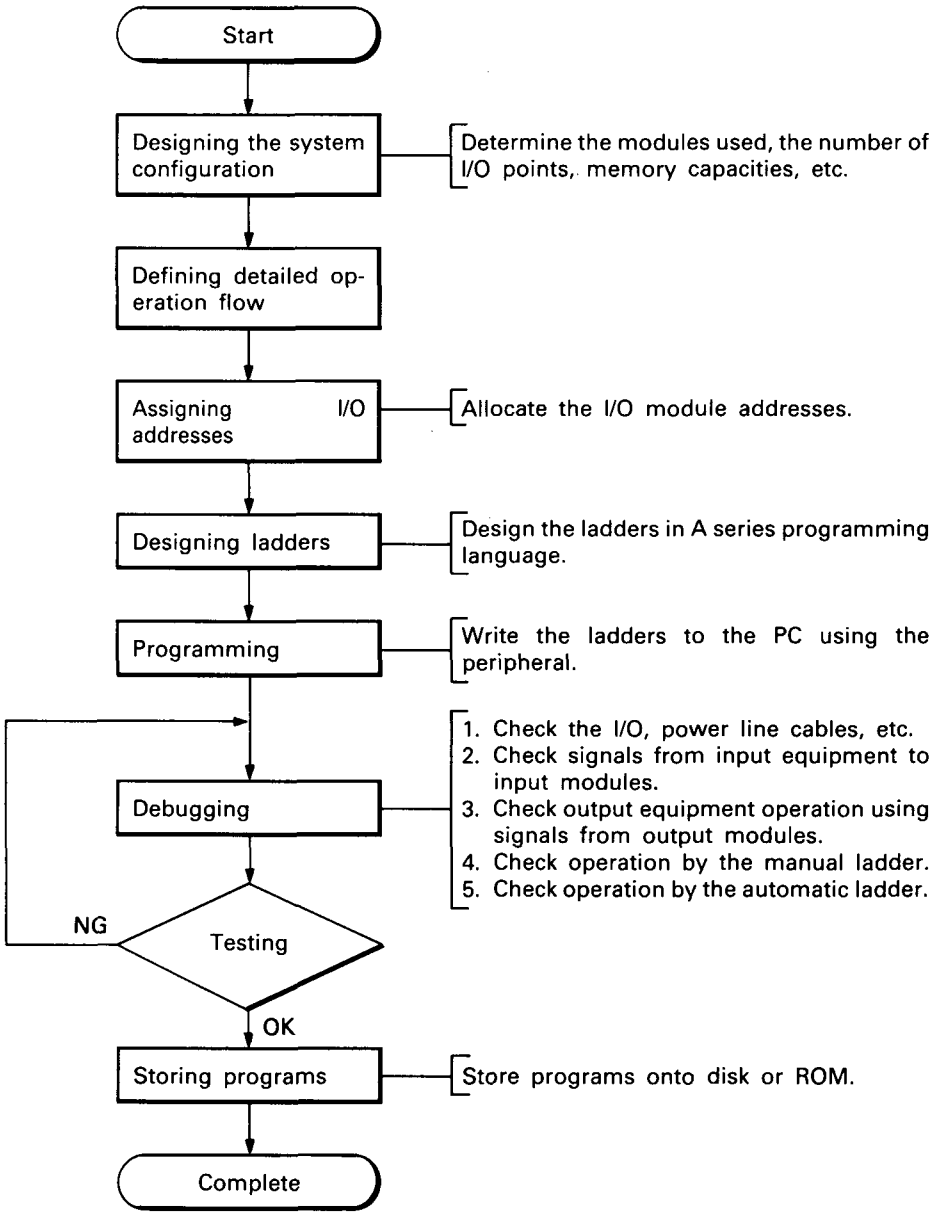
Example:

Alphanumeric Character	ASCII Code
0	30
1	31
2	32
A	41
B	42
C	43

2) For further details, see Appendix 4.

3.3 System Designing Procedure

When using the PC, any system should be designed in the following procedure which is basically the same as that of the relay control panel designing.



IMPORTANT

Note the following:
On switching on the PC power supply, there is a short but finite time before the DC levels reach their operating values. During this period, the unit will not operate normally. The same applies when the power is cut as the DC levels drop below their operating values. Make up a circuit which will overcome any problems which may arise in output control due to this phenomenon.

3.4 User Memory Configuration

The user memory may be arranged as required to suit the individual application. Memory allocation is made using any of the peripheral programming devices. Where memory allocation has not been made, the PC uses its default settings as described in Section 3.4.1.

The ACPU allows the following programs to be stored in the sequence program area.

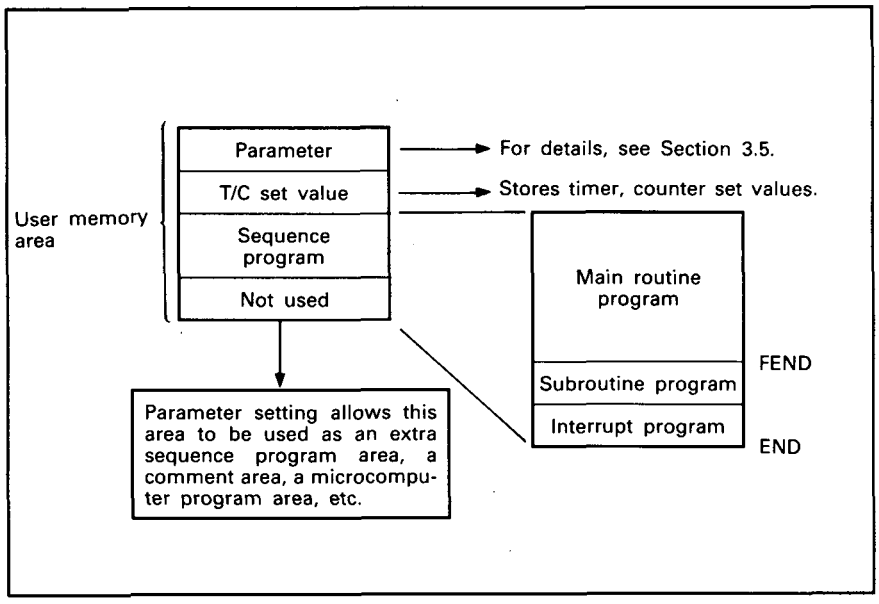


Fig. 3.3 Sequence Program Area Configuration

3.4.1 User memory not assigned by parameter setting

(1) A1NCPU

The following memory map is the default settings adopted by the A1NCPU when parameter setting has not been made.

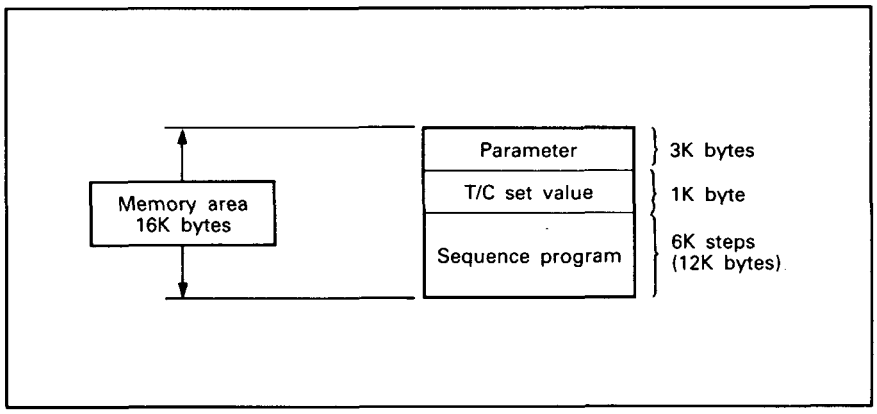


Fig. 3.4 Memory Area Configuration

(2) A2NCPU

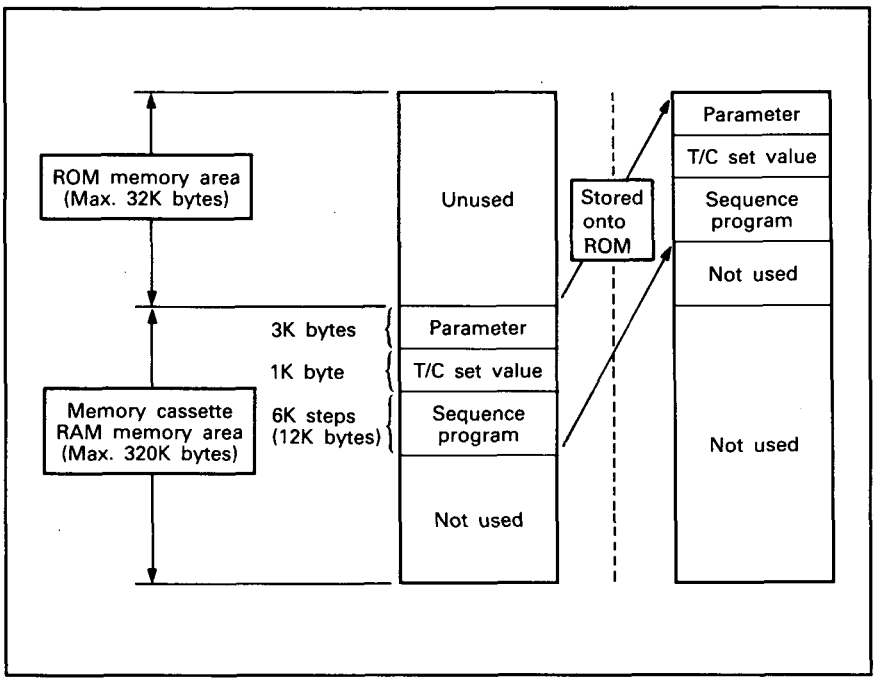


Fig. 3.5 Memory Area Configuration

(3) A3N, A3HCPU

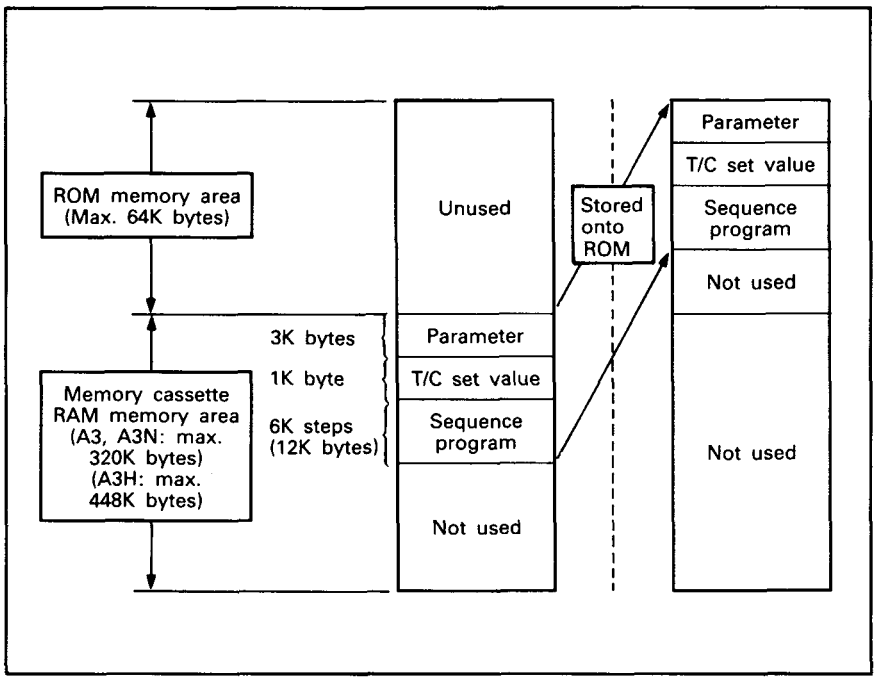


Fig. 3.6 Memory Area Configuration

3.4.2 User memory assigned in parameters

(1) A1NCPU

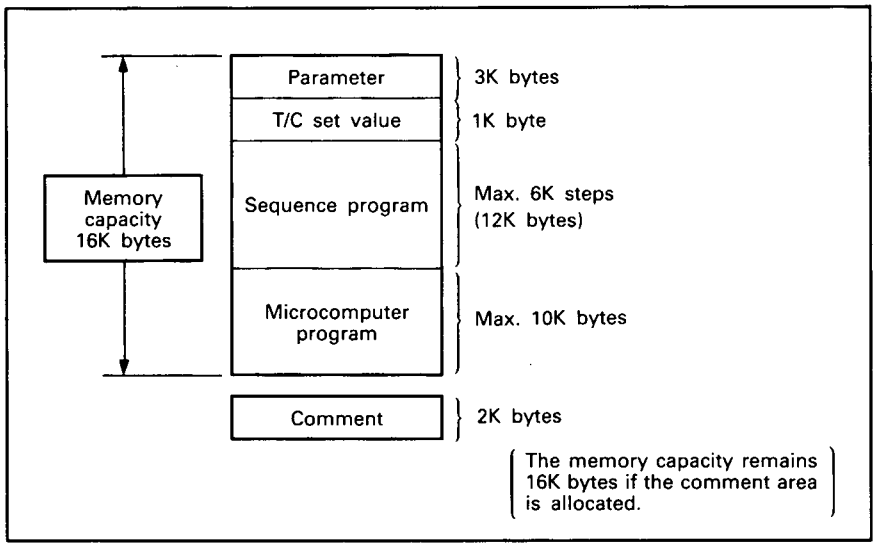


Fig. 3.7 Memory Area Configuration

(2) A2NCPU

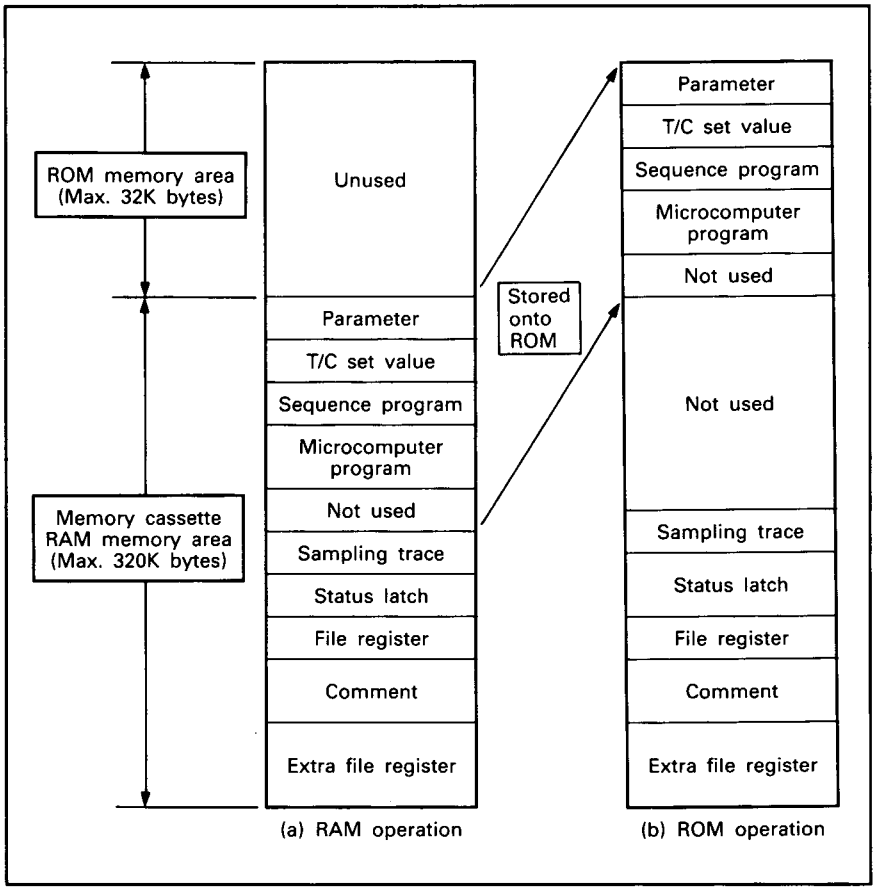


Fig. 3.8 Memory Area Configuration

(3) A3NCPU

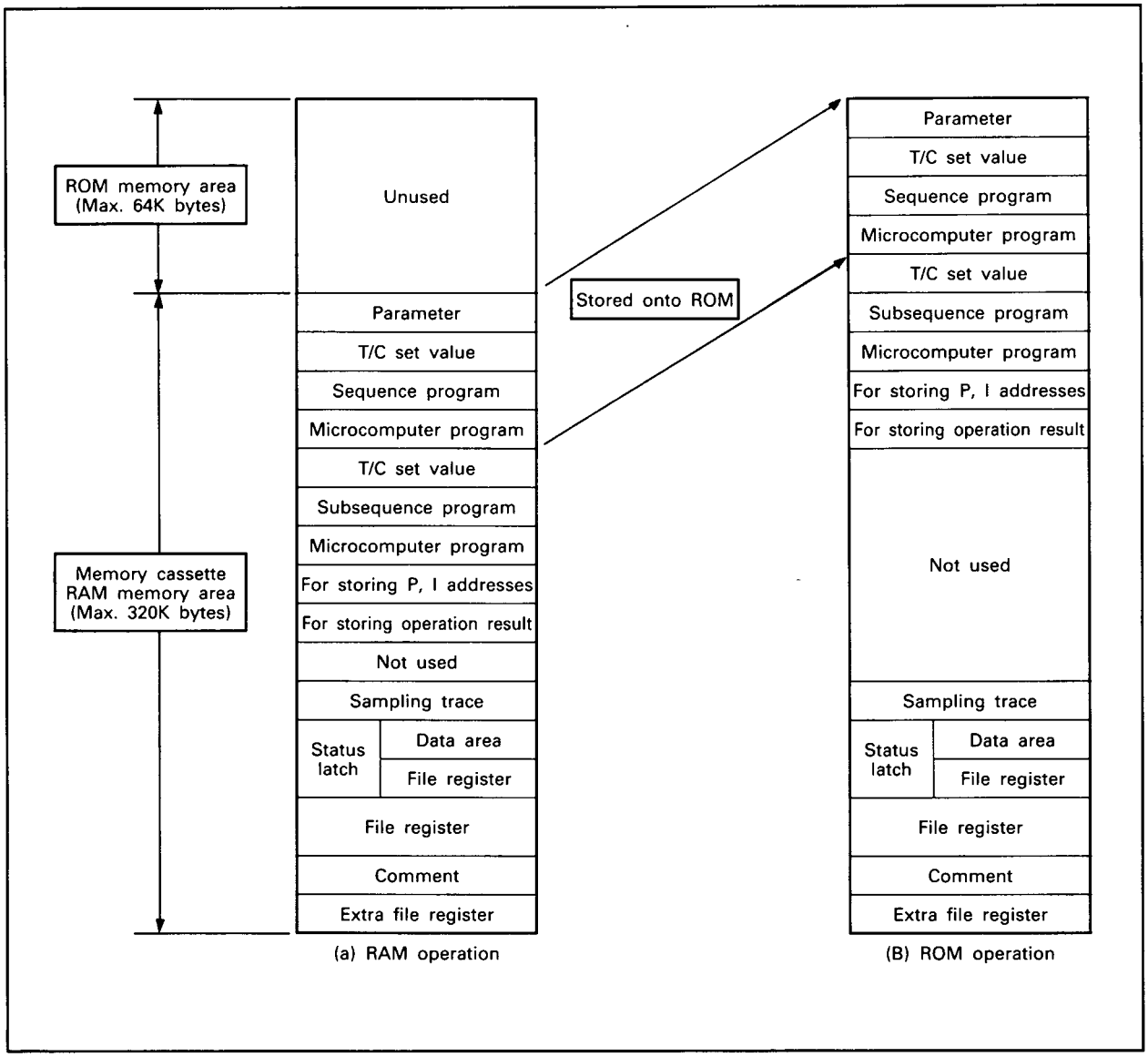


Fig. 3.9 Memory Area Configuration

(4) A3HCPU

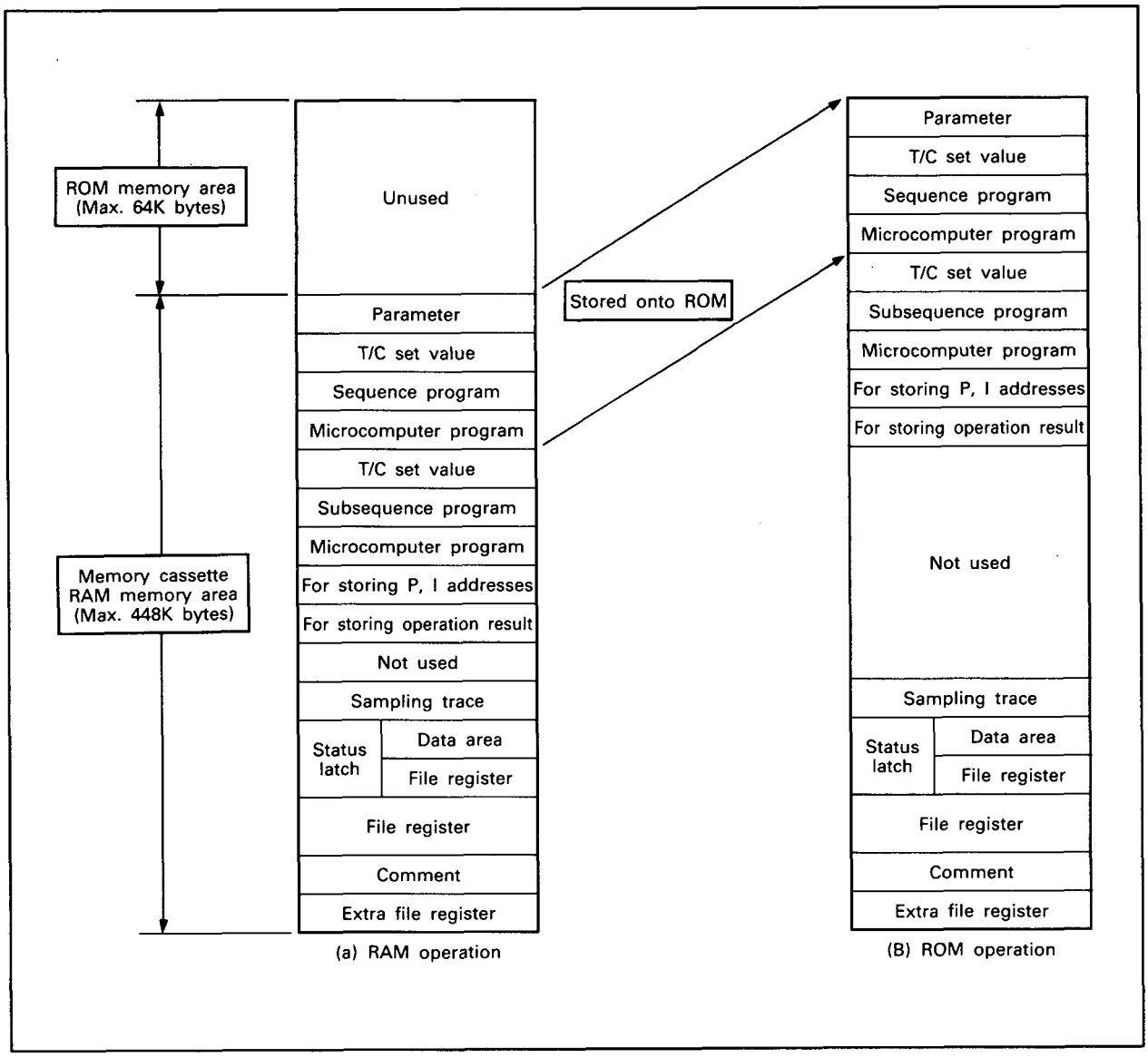


Fig. 3.10 Memory Area Configuration

(5) The memory area assigned in parameters to the A2N, A3N, and A3HCPU is 144K bytes maximum. Use the extra file register area when more than 144K bytes are required.

(6) The extra file register area may be accessed by storing utility program SW0GHP-UTLPC-FN1 into the microcomputer program area and calling this program from the sequence program by the SUB instruction.
For further details, see the SW0GHP-UTLPC-FN1 Operating Manual.

3.5 Parameter Setting

Default parameter values may be selected or user parameters may be written using any of the peripheral devices. Table 3.1 gives details on parameter setting.

REMARKS

Parameter setting involves specifying various PC functions and device ranges as well as assigning the user memory. The set data is stored in the parameter memory area (the first 3K bytes of the user memory area).

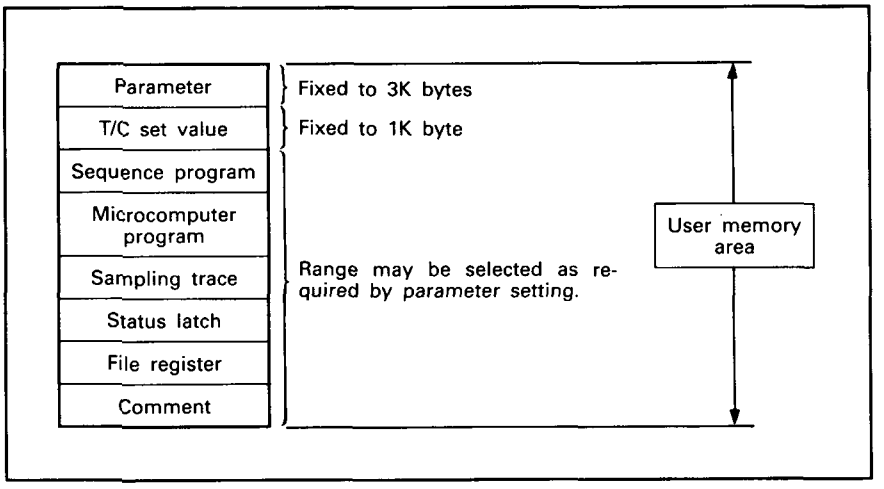


Fig. 3.11 User Memory Area Configuration

3.5.1 Parameter setting ranges

Item \ CPU / Setting			A1NCPU		A2NCPU	
			Default value	Setting range	Default value	Setting range
User memory area assignment	Sequence program memory capacity		6K steps	1 to 6K steps (1K step increments)	6K steps	1 to 14K steps (1K step increments)
	Subsequence program memory capacity					
	File register capacity				None	1 to 4K points (1K point increments)
	Comment capacity		None	128 points (128 point increments)	None	0 to 4032 points (64 point increments)
	Status latch	Memory capacity			None	0/8 to 16K bytes
		Data memory				Yes/no
		File register				Yes/no (2 to 8K bytes)
	Sampling trace	Memory capacity			None	0/8K bytes
		Device setting				Device number
		Execution condition				Scan specification
						Time specification
		Sampling count				0 to 1024 times (128 time increments)
	Microcomputer program capacity		None	0 to 10K bytes (2K byte increments)	None	0 to 26K bytes (2K byte increments)
I/O control setting			Both input, output in direct mode	Direct or refresh mode can be specified separately for input and output	Both input, output in direct mode	Direct or refresh mode can be specified separately for input and output
Latch range setting	Link relay (B)		L1000 to 2047 only No setting for others.	B0 to 3FF (1 point increments)	L1000 to 2047 only No setting for others.	B0 to 3FF (1 point increments)
	Timer (T)			T0 to 255 (1 point increments)		T0 to 255 (1 point increments)
	Counter (C)			C0 to 255 (1 point increments)		C0 to 255 (1 point increments)
	Data register (D)			D0 to 1023 (1 point increments)		D0 to 1023 (1 point increments)
	Link register (W)			W0 to 3FF (1 point increments)		W0 to 3FF (1 point increments)
Link range setting	Number of link stations		None	1 to 64	None	1 to 64
	Input (X)			X0 to FF (16 point increments)		X0 to 1FF (16 point increments)
	Output (Y)			Y0 to FF (16 point increments)		Y0 to 1FF (16 point increments)
	Link relay (B)			B0 to 3FF (16 point increments)		B0 to 3FF (16 point increments)
	Link register (W)			W0 to 3FF (1 point increments)		W0 to 3FF (1 point increments)
Internal relay (M), latch relay (L), step relay (S) setting			M0 to 999 L1000 to 2047	M/L/S0 to 2047 M, L, S in serial numbers to be set in order of M, L, S	M0 to 999 L1000 to 2047	M/L/S0 to 2047 M, L, S in serial numbers to be set in order of M, L, S
Watch dog timer setting			200ms	10ms to 2000ms (10ms increments)	200ms	10ms to 2000ms (10ms increments)
Timer setting			100ms: T0 to 199 10ms: T200 to 255	100ms + 10ms + retentive timers = 256 points (8 point increments) Timers in serial numbers to be set in order of 100ms, 10ms, retentive	100ms: T0 to 199 10ms: T200 to 255	100ms + 10ms + retentive timers = 256 points (8 point increments) Timers in serial numbers to be set in order of 100ms, 10ms, retentive

Table 3.1 Parameter Setting Ranges (Continue)

A3N, A3HCPU		Remarks	Usable Peripheral	
Default value	Setting range		PU	GPP HGP PHP
6K steps	1 to 30K steps (1K step increments)		○	○
None	1 to 30K steps (1K step increments)		○	○
None	1 to 8K points (1K point increments)		○	○
None	0 to 4032 points (64 point increments)		–	○
None	0/8 to 24K bytes		–	○
	Yes/no			
	Yes/no (2 to 16K bytes)			
None	0/8K bytes		–	○
	Device number			
	Scan specification			
	Time specification			
	0 to 1024 times (128 time increments)			
None	0 to 58K bytes (2K byte increments)		–	○
Both input, output in direct mode	Direct or refresh mode can be specified separately for input and output		○	○
L1000 to 2047 only No setting for others.	B0 to 3FF (1 point increments)		○	○
	T0 to 255 (1 point increments)			
	C0 to 255 (1 point increments)			
	D0 to 1023 (1 point increments)			
	W0 to 3FF (1 point increments)			
None	1 to 64		–	○
	X0 to 7FF (16 point increments)			
	Y0 to 7FF (16 point increments)			
	B0 to 3FF (16 point increments)			
	W0 to 3FF (1 point increments)			
M0 to 999 L1000 to 2047	M/L/S0 to 2047 M, L, S in serial numbers to be set in order of M, L, S		○	○
200ms	A3N: 10ms to 2000ms (10ms increments) A3H: 200ms only		○	○
100ms: T0 to 199 10ms: T200 to 255	100ms + 10ms + retentive timers = 256 points (8 point increments) Timers in serial numbers to be set in order of 100ms, 10ms, retentive		○	○

Table 3.1 Parameter Setting Ranges (Continue)

Item \ CPU / Setting		A1NCPU		A2NCPU	
		Default value	Setting range	Default value	Setting range
Counter setting		Without interrupt counter	Counter + interrupt counter = 256 points (8 point increments) Counters in serial numbers	Without interrupt counter	Counter + interrupt counter = 256 points (8 point increments) Counters in serial numbers
I/O number assignment	Input (X) module	None	0 to 64 points (16 point increments)	None	0 to 64 points (16 point increments)
	Output (Y) module				
	Special function module				
	Vacant slot				
Remote run/pause control from digital input		None	X0 to FF (1 point each for run and pause contacts. Setting of pause contact alone is not allowed.)	None	X0 to 1FF (1 point each for run and pause contacts. Setting of pause contact alone is not allowed.)
Operation mode at time of error	Fuse blow	Continue	Stop/continue	Continue	Stop/continue
	I/O verify error	Stop		Stop	
	Operation error	Continue		Continue	
	Special function module check error	Stop		Stop	
Annunciator display mode					
STOP → RUN operating mode		Operating status prior to stop is re-output.	Output data may be re-used at beginning of new operation or cleared	Operating status prior to stop is re-output.	Output data may be re-used at beginning of new operation or cleared
Print title		None	An alphanumeric print out title may be added.	None	An alphanumeric print out title may be added.
Keyword entry		None	Max. 6 digits in hexadecimal (0 to 9, A to F)	None	Max. 6 digits in hexadecimal (0 to 9, A to F)

Table 3.1 Parameter Setting Ranges (Continue)

A3N, A3HCPU		Remarks	Usable Peripheral	
Default value	Setting range		PU	GPP HGP PHP
Without interrupt counter	A3N: Counter + interrupt counter = 256 points (8 point increments) Counters in serial numbers A3H: C224 to 255 used to count the number of interrupt signals by parameter setting		–	○
None	0 to 64 points (16 point increments)		–	○
None	X0 to 7FF (1 point each for run and pause contacts. Setting of pause contact alone is not allowed.)		–	○
Continue	Stop/continue		–	○
Stop				
Continue				
Stop				
F number display	Display of only F number or alternate display of F number and comment		–	○
Operating status prior to stop is re-output.	Output data may be re-used at beginning of new operation or cleared		–	○
None	An alphanumeric print out title may be added.		–	○
None	Max. 6 digits in hexadecimal (0 to 9, A to F)		○	○

Table 3.1 Parameter Setting Ranges

REMARKS

(1) When estimating the memory cassette size required, calculate the number of bytes used from the settings made as follows:

Item	Setting Unit	Number of Bytes
Sequence program capacity	1K step	2K bytes
Subsequence program capacity		
File register capacity	1K point	2K bytes
Comment capacity	64 points	1K byte
Sampling trace enabled	128 times	1K byte

(2) The minimum comment capacity setting is 1K byte.

3.5.2 Main program memory capacity

1) The main program area is divided into the sequence program and microcomputer program areas.

Main program memory capacity	=	Sequence program memory capacity	+	Microcomputer program memory capacity

2) The default value is only available for the sequence program memory capacity (6K steps).

3) Parameters may be set for the sequence program memory capacity only or the sequence program + microcomputer program memory capacity.

POINT

The minimum sequence program capacity required for the sequence program is 1K step to execute the microcomputer program as the microcomputer program is called by the SUB instruction in the sequence program.

3.5.3 Subprogram memory capacity

1) The subprogram area is divided into the sequence program and microcomputer program areas.

Subprogram memory capacity	=	Sequence program memory capacity	+	Microcomputer program memory capacity

2) Parameters may be set for the sequence program memory capacity only or the sequence program + microcomputer program memory capacity.

3) With the sequence programs and microcomputer programs written in the main and subprogram areas, respectively, two programs (main program and subprogram) can be run alternately in series or either program can be separately selected and run.

POINT

(1) The minimum sequence program capacity required for the sequence program is 1K step to execute the microcomputer program as the microcomputer program is called by the SUB instruction in the sequence program.

(2) By setting the subprogram capacity, the timer/counter set value area and P, I address storage area are automatically set to 6K bytes.

3.5.4 File register

1) Assigned to the end of the user memory area.

2) When extra data registers are required, the user memory area is used as file registers.

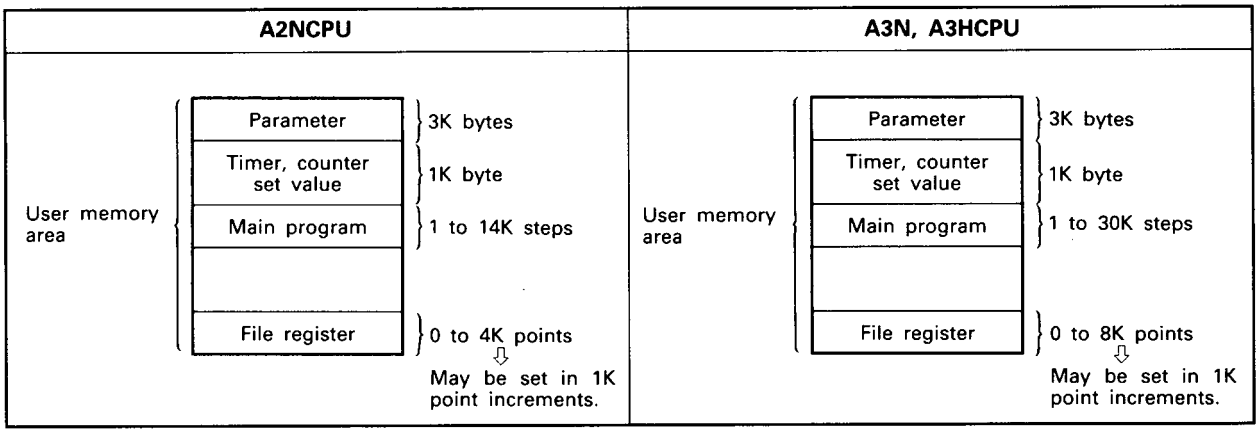


Fig. 3.12 File Register Assignment

3.5.5 Comment

1) Can be set to 128 points (2K bytes), F0 to 127, when the A1NCPU is used. Can be set to a maximum of 4032 points in 64 point (1K byte) increments when the A2N, A3N, A3HCPU is used.

3) The comment written is stored in the PC as follows:

A1NCPU·····Memory area in the CPU module.
A2NCPU, A3H, A3NCPU ·····User memory area in the memory cassette.

3.5.6 Status latch

1) Used as a fault finding facility, this enables a "snap shot" of the PC device memory to be taken and stored in a dedicated area of the PC memory. The snap shot is triggered by the SLT instruction. This allows the device memory to be monitored and is very useful for program debugging.

2) The following data may be written to the status latch memory area:
   - Data memories

   X, Y, M, L, S, B, F, T/C (contact, coil)·····ON/OFF data
   T, C·····Present value
   D, W, A0, A1, Z, V·····Data

   - File register·····Data

3) The status latch area is set as follows:

Data memory	8K bytes
File register	Number of points set

3.5.7 Sampling trace

1) Used as a fault finding facility, this allows the data from a selection of specified devices to be recorded in a dedicated area of the PC memory for each of a defined number of scans or at defined time intervals. This allows the recorded progress of device statuses to be examined over a series of scans or time intervals and is very useful for program debugging.

2) The following data may be set to the sampling trace area:

   (a) Parameter setting

   - Memory capacity

   (b) List monitoring

   - Device
   - Execution condition
   - Sampling count after [STRA] instruction execution
   - Total sampling count

3) Memory capacity setting

   Set whether or not the sampling trace function is executed. 8K bytes are set by specifying the sampling trace.

4) Device setting

   The following devices may be used for the sampling trace:

Device		Max. Number of Points
Bit device	X, Y, M, L, S, B, F, T/C coil, T/C contact	8
Word device	T, C, D, W, R, A, Z, V	3

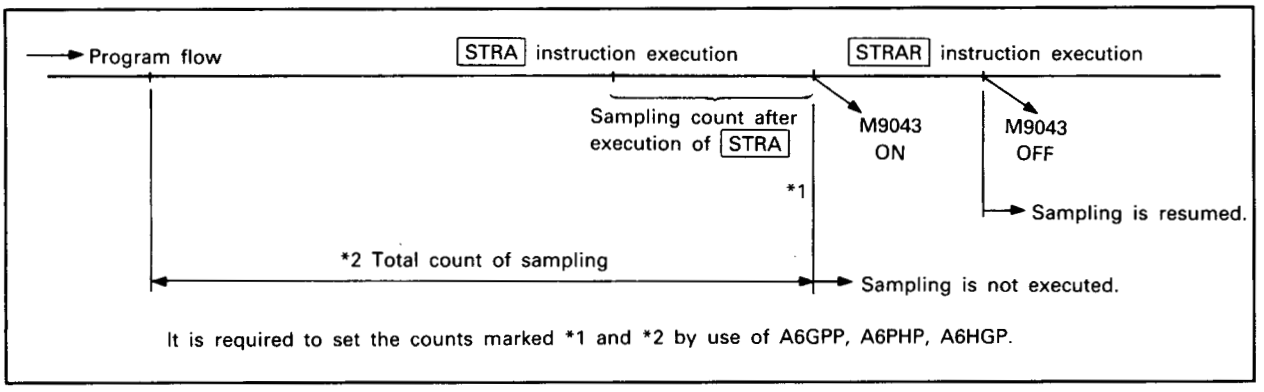
5) Execution condition setting

Sampling trace may be executed by either of the following conditions:

Per [END].
Per specified time interval·····Time specified

6) Sampling data stop and resumption methods

After sampling is performed at the count preset on the A6GPP, A6PHP, A6HGP after the [STRA] instruction is executed, the sampling trace data is latched and the sampling is stopped.
To resume the sampling, execute the [STRAR] instruction.

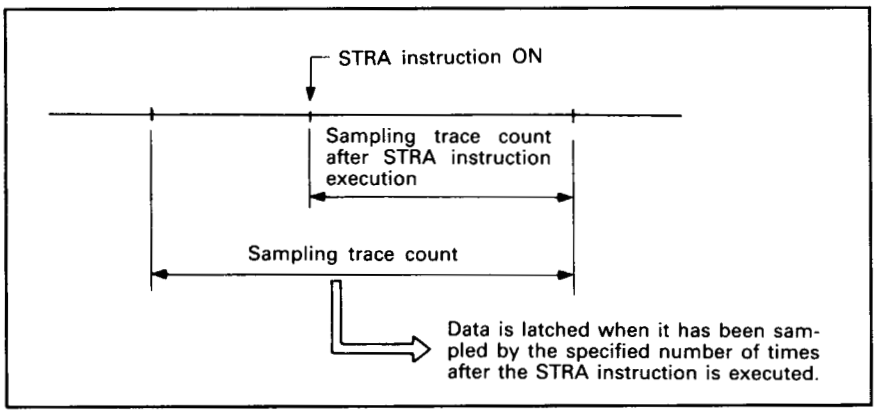


The sampling count after [STRA] instruction execution must satisfy the following condition:

$$\left[\begin{array}{c}\text{Sampling count after} \\ \text{[STRA] instruction} \\ \text{execution}\end{array}\right] \leqq \left[\text{total sampling count}\right]$$

7) Total sampling count

May be set to 1024 times max.



3.5.8 Latch range setting

1) Any latched device data remains unchanged if power failure occurs during PC run and is not cleared when power is restored or the RESET switch set to RESET.
2) The latched device data is cleared by setting the RESET switch to LATCH CLEAR with the RUN key switch in STOP position.

3.5.9 Link range setting

Set the link range as follows in accordance with the devices used:

- Input (X), output (Y) ····· Range for use in the master, local and remote stations.
- Link relay ····· Range in which the link relay is switched on/off as a coil in the master and local stations.
- Link register ····· Data storage range in the master and local stations.

3.5.10 Internal relay (M), latch relay (L), step relay (S) setting

Must be allocated in order of the internal, latch and step relays and may be set independently, i.e. internal relay only, latch relay only or step relay only.

3.5.11 Watch dog timer setting

Must be equal to or greater than the maximum value of scan time. (The A3HCPU setting is fixed at 200ms.)

3.5.12 Timer selection

1) Specify the head number of each timer. In this case, the 10ms timer head number must be less than that of the 100ms retentive timer.
2) Timer setting may be made in multiples of 8 points and must be assigned in order of 100ms, 10ms and 100ms retentive timers.

3.5.13 Counter selection

1) Specify the head number of the interrupt counter in units of 8 points.
2) Must be assigned in order of the counter and interrupt counter, and may be specified independently, i.e. counter only or interrupt counter only.
3) The A3H interrupt counters are used to count the number of interrupt signals and C224 to 255 are used as required.

3.5.14 I/O assignment

1) Specify the number of points for the main and extension base slots in multiples of 16 points.

2) The parameter I/O assignment has priority over the actual module's I/O capacity.

3.5.15 Remote run/pause contact

1) The remote run/pause input contacts allow the PC to be switched to STOP/PAUSE mode.

2) 1 point can be set to each of the remote run and remote pause contacts. Setting of the pause contact alone is not allowed.

3.5.16 Operation mode at time of error

Specify whether the PC operation is continued or stopped when an error has occurred.

Error	CPU Status					
	Operation	RUN LED		Special relay switched on	Special register for storing data	Self-check error number (D9008)
	Default value	A1N, A2N	A3N, A3H			
Operation error	Sequence program error, e.g. the value to be converted into BCD is greater than 0 to 9999 (or 0 to 99999999).	Continue	Flicker	On	M9010 M9011	D9010 D9011	50
I/O module verify error	Any I/O module status detected is different from that at power on (e.g. 32-point module change).	Stop	Flicker	Off	M9002	D9002	31
Fuse blow error	An output module fuse has blown.	Continue	Flicker	On	M9000	D9000	32
Function module error	FROM/TO Instruction has been executed to the slot without any special function module.	Stop	Flicker	Off	M9010 M9011	D9010 D9011	46

Table 3.2 Operation Mode at Time of Error

3.5.17 Annunciator display mode

1) Specify whether a comment is indicated or not in addition to the F number detected, on the CPU front LED display.

2) No comment·····Only the F number is displayed.
Comment displayed·····The F number and comment are displayed alternately at intervals of 2 seconds.

3.5.18 STOP to RUN operating mode

1) Specify the output Y status when the RUN key switch is set from STOP to RUN.

2) Re-output·····Operation status prior to stop is re-output.
Output after operation execution·····Operation status prior to stop is cleared and status at the beginning of new operation is output.

3.5.19 Entry code

The entry code is made to protect the contents of program of the programmable controller. Enter the entry code to a maximum of 6 digits by use of 0 to 9 and A to F alphanumeric characters. If the entry code has been entered, the read operation of parameters, main sequence program and sub-sequence program in the programmable controller and the write operation from the peripheral equipment cannot be performed unless the entry code is entered on the peripheral equipment.

The above described read and write operations cannot be performed if a different entry code is set. The entry code can be canceled or changed after the code is set and the operation is ready. However, the entered code cannot be read.

If you have forgotten the entry code, the operation of programmable controller cannot be performed until the all clear operation by the A7PU or A6GPP, A6HGP is performed to clear all the contents of memory. In this case, all the stored memory contents of programmable controller are erased. For the details of entry code loading procedure, refer to the Operation Manual for A7PU or A6GPP, A6HGP.

3

3.5.20 Print title entry

By use of the print title entry function, names and machine names can be provided for programs created by the user and stored in the user memory area of PC.

Also, a print title can be printed and can be used for the cover of user program.

A maximum of 128 characters can be entered and all the keys on the MELSAP keyboard can be used for the entry.

For the details of print title describing and printing procedures, refer to the Operation Manual for A6GPP, A6HGP (A series).

3.6 CPU Processing

3.6.1 Operation processing

(1) Stored program

The user-written program stored in the PC CPU memory area is executed by the CPU reading and acting on its instructions. Device statuses are controlled in accordance with the operation result.

(2) Repeated operation

A series of processing operations executed repeatedly as follows:
The PC sequentially executes the program stored in the CPU memory area beginning with step 0. After the END (FEND) instruction is executed, the PC performs internal processing (e.g. timer/counter present value update, self-diagnosis) and returns to step 0.

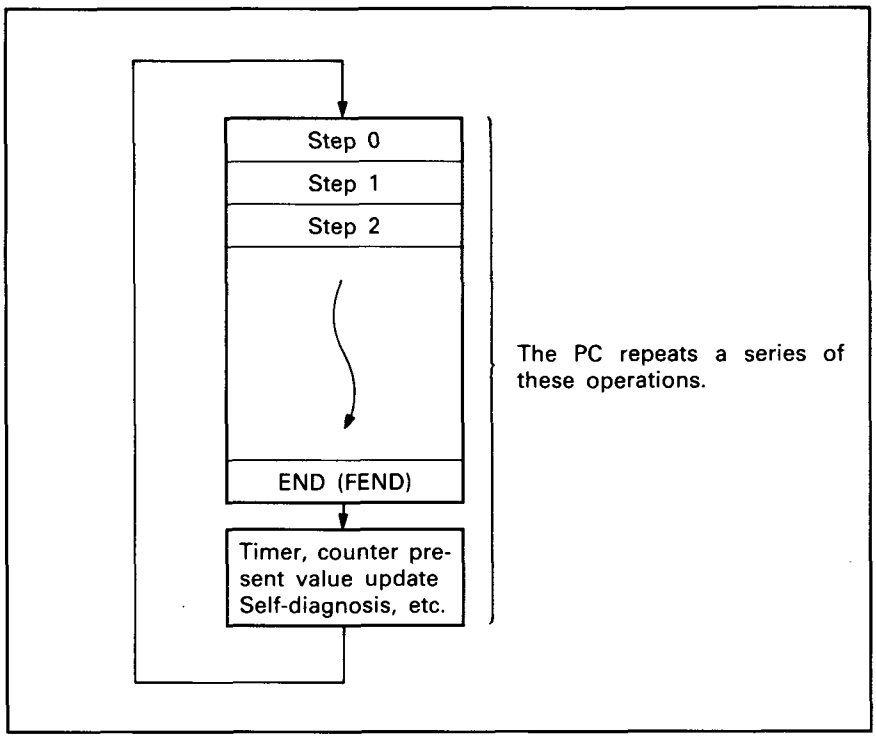


Fig. 3.13 PC Operation Processing

REMARKS

Processing from one step 0 to another or one END (FEND) to another is referred to as 1 scan. Hence, 1 scan of the PC is the sum of the user-written program (step 0 to END) processing time and PC internal processing time.

3.6.2 I/O processing

In the MELSEC-A series, the I/O processing system depends on the CPU as follows:

A1N, A2N, A3N, A3H·····Direct or refresh mode selected

(1) Direct mode

1) In this mode, each input signal is entered to the CPU and used as input data. Each operation result in the program is output to the output data memory and output module.

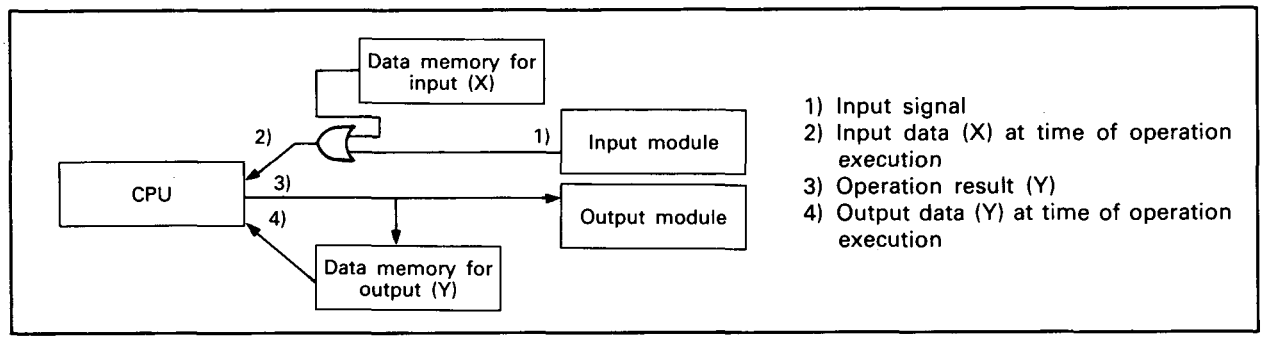
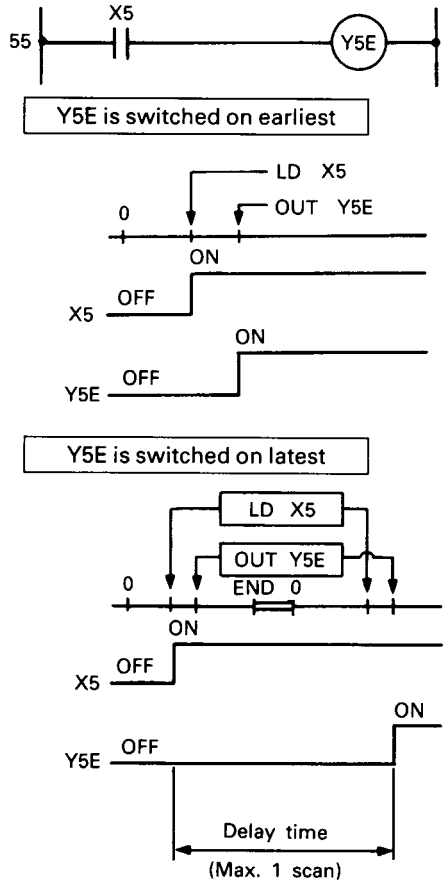


Fig. 3.14 I/O Data Flow in Direct Mode

2) An output module change lags max. 1 scan behind the corresponding input module change as shown in Fig. 3.15.

Ladder example



Output Y5E is switched on by input X5 switched on.

Input X5 is switched on immediately before the operation of step 55 is executed. Y5E is switched on during the scan in which X5 has been switched on. The delay time is therefore minimal.

Input X5 is switched on after the operation of step 55 is executed. Y5E is switched on during the next scan. The delay time is therefore 1 scan max.

Fig. 3.15 Output Y Change to Corresponding Input X Change

(2) Refresh mode

1) In this mode, a batch of input module changes is entered to the input data memory of the CPU before each scan is executed. This input data memory data is used to execute operation.
Each operation result is output to the output data memory. After the END instruction is executed, a batch of the output data memory contents is output to the output module.

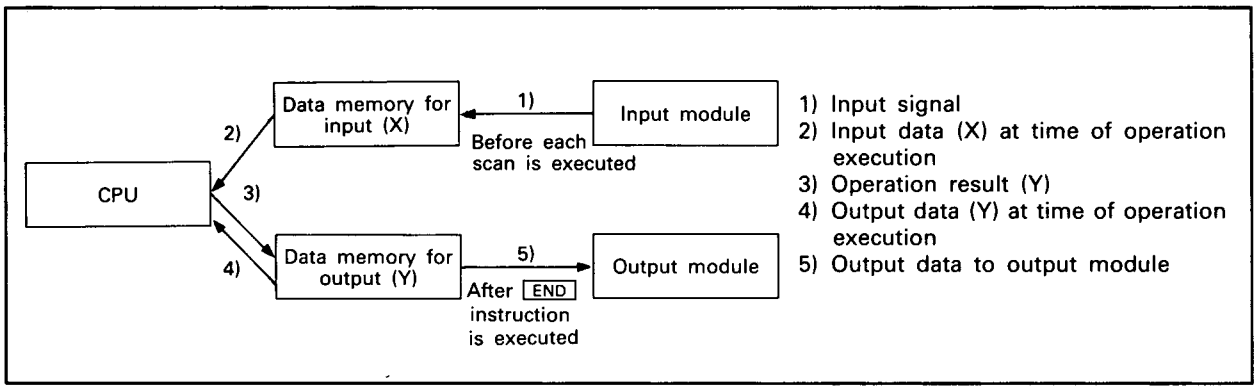
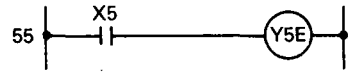


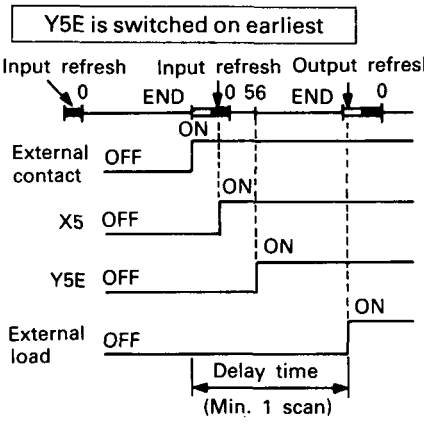
Fig. 3.16 I/O Data Flow in Refresh Mode

2) An output module change lags max. 2 scans behind the corresponding input module change as shown in Fig. 3.17. Note that 1 scan time in refresh mode is less than that in direct mode if the I/O refresh time is included because the processing time of one instruction is shorter in refresh mode than in direct mode.

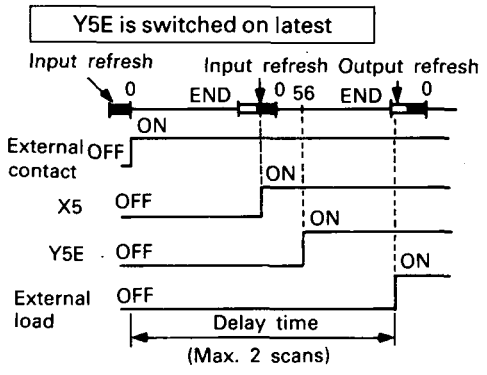
Ladder example



Output Y5E is switched on by input X5 switched on.



The external contacts close immediately before the input refresh is made. X5 is switched on when the input refresh is made. Y5E is switched on when the operation of step 56 is executed. The external load is switched on when the output refresh is made after the END instruction is executed. The delay time (external contact change to external load change) is therefore 1 scan.



The external contacts close immediately after the input refresh is made. X5 is switched on when the next input refresh is made. Y5E is switched on when the operation of step 56 is executed. The external load is switched on when the output refresh is made after the END instruction is executed. The delay time (external contact change to external load change) is therefore 2 scans.

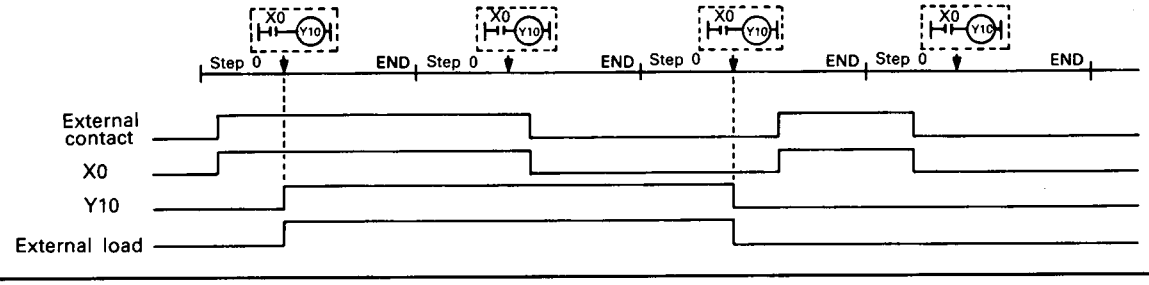
Fig. 3.17 Output Y Change to Corresponding Input X Change

(3) I/O timing comparison

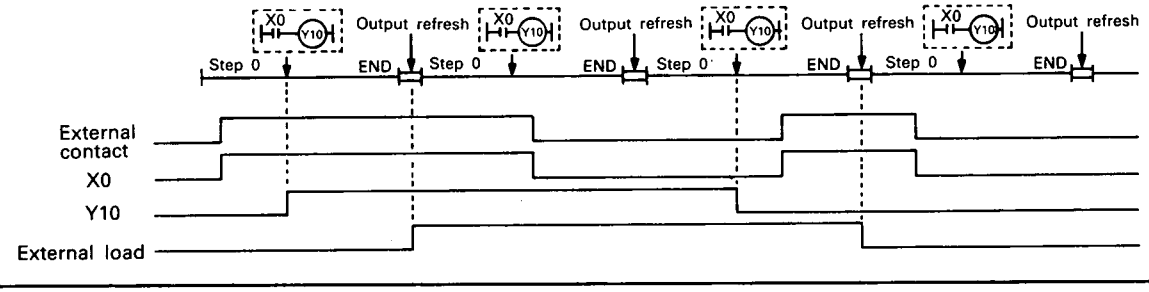
The ON/OFF timings of input (X), output (Y) and external load differ as shown below in direct and refresh modes.

I/O Control Mode		CPU Used			
Input	Output	A1N	A2N	A3N	A3H
Direct	Direct	○	○	○	○
Direct	Refresh				○
Refresh	Direct	○	○	○	○
Refresh	Refresh	○	○	○	○

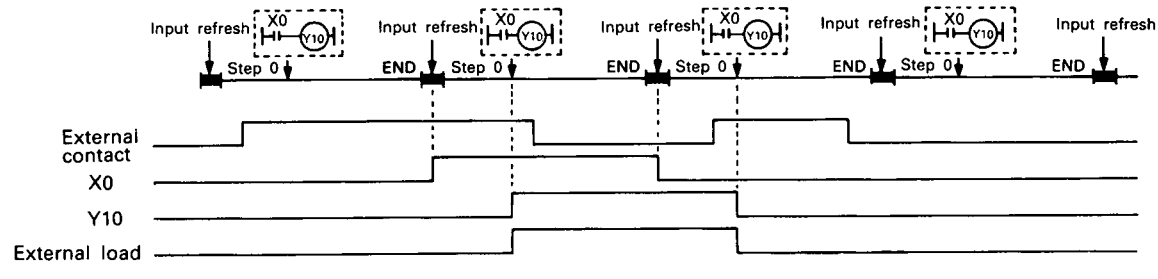
Input: direct, output: direct



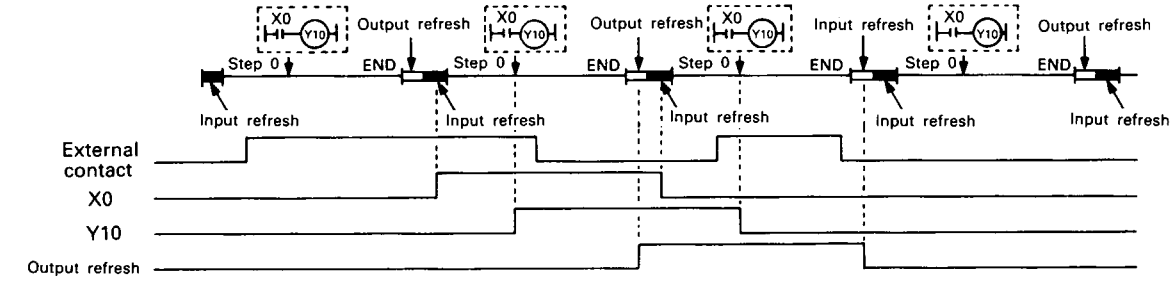
Input: direct, output: refresh



Input: refresh, output: direct



Input: refresh, output: refresh



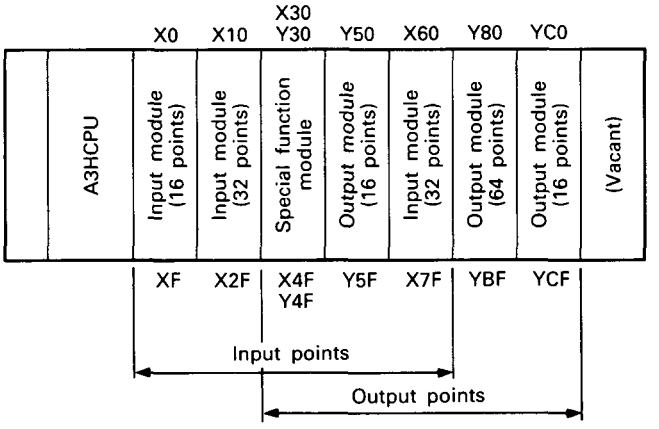
(4) I/O refresh time

1) Input refresh (before execution of step 0) and output refresh (after execution of END instruction) processing time can be calculated as follows:

$$\text{Refresh time} = \frac{\text{number of input + output points}}{16} \times t \ [\mu\text{sec}]$$

where, t = 5.4 (A1N to A3N) or 4.375 (A3H)

2) The number of input and output points depends on the I/O module and special function module locations as shown below:



POINT

The special function module is calculated as having 32 inputs and 32 outputs.

Example: X30 to X4F, Y30 to Y4F

3) The number of input points is the sum of the I/O points of all modules from the input or special function module in the lowest slot number to the one in the highest slot number.

Example: Sum of the module points between X0 and X7F in the above I/O allocation

Input points = 16 + 32 + 32 + 16 + 32 = 128

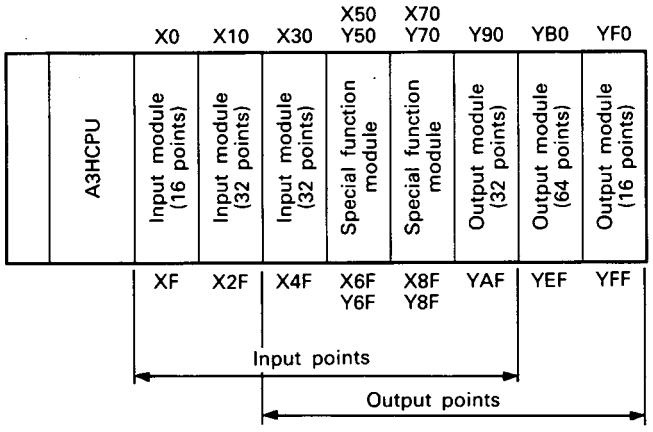
4) The number of output points is the sum of the I/O points of all modules from the output or special function module in the lowest slot number to the one in the highest slot number.

Example: Sum of the module points between Y30 and YCF in the above I/O allocation

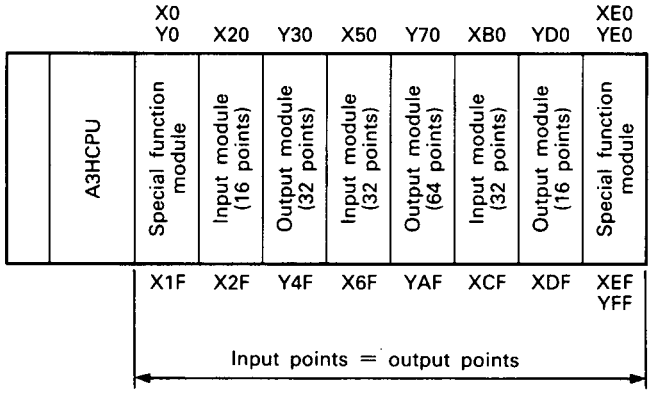
Output points = 32 + 16 + 32 + 64 + 16 = 160

5) Calculation of refresh time

Refresh time can be reduced by arranging the same modules sequentially as shown on the left below.



Same modules loaded sequentially



Same modules loaded non-sequentially

3.6.3 Scan time

1) Indicates the execution time of 1 scan.

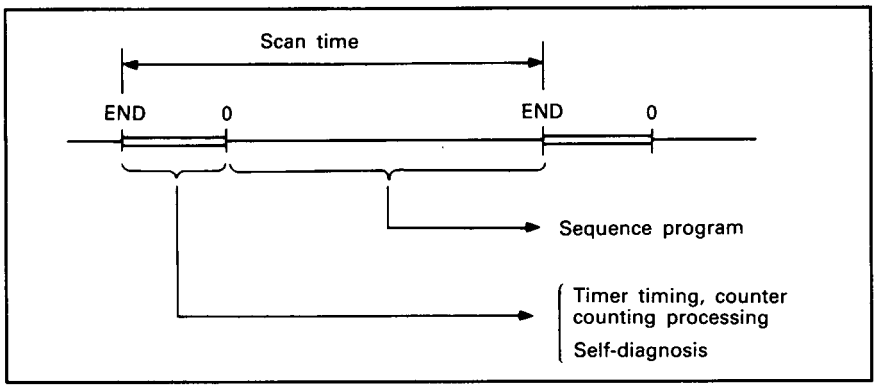


Fig. 3.18 Scan Time

2) Scan time of the ACPU used in an independent system can be calculated as follows.

$$\text{(Scan time)} = \left\{\begin{array}{c}\text{sum of instruction}\\ \text{operation times}\end{array}\right\} + \left\{\begin{array}{c}\text{timer timing, counter counting}\\ \text{processing and self-diagnosis time}\end{array}\right\}$$

REMARKS

An independent system consists of a main base and extension base(s) connected by extension cables and is not used with data link or computer link.

3) The PC stores scan times in special registers D9017 to 9019 in units of 10ms.

a) Data stored in D9017 to 9019

- D9017·····Minimum value of scan time
- D9018·····Present value of scan time
- D9019·····Maximum value of scan time

b) Scan time accuracy

Scan time accuracy is ± 10ms.
Hence, when 5 exists in D9017 to 9019, the actual scan time is between 40 and 60ms.

c) D9017 to 9019 are not cleared and store the scan time if the WDT instruction is executed.

REMARKS

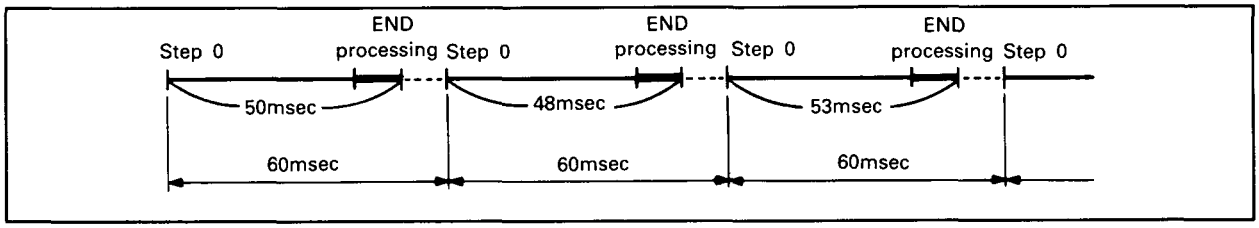
Scan time monitored by the peripheral is as follows:

- Scan time is 0 to 20ms·····10ms is indicated.
- Scan time is 10 to 30ms·····20ms is indicated.

3.6.4 Constant scan

(1) Constant scan

Executes the program repeatedly at a specified interval as shown below.



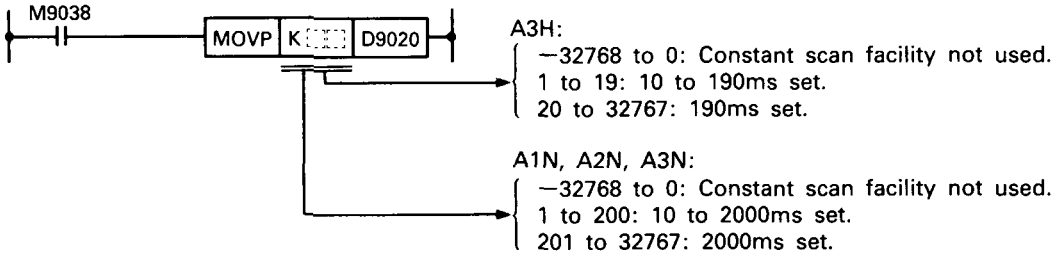
(2) Constant scan time setting

1) Setting range is as follows:

A1N, A2N, A3NCPU·····10ms to 2000ms in 10ms increments

A3HCPU·····10ms to 190ms in 10ms increments

2) The set value is written to special data register D9020.

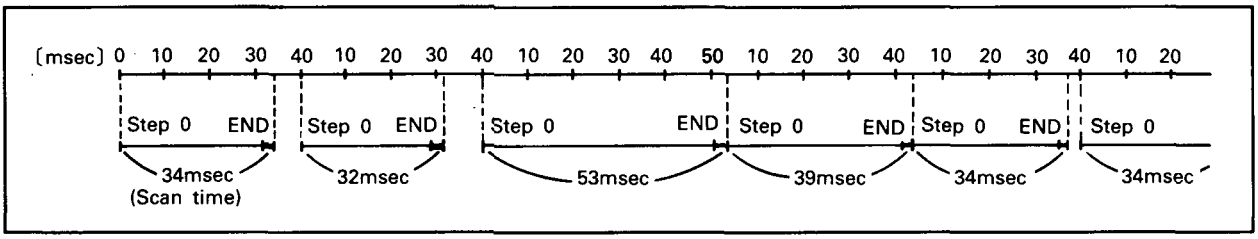


POINT

The ladder rung shown above should be written at the beginning of the sequence program since D9020 is cleared to 0 when the PC is switched on or reset.

3) The constant scan period is ignored if the sequence program scan time is greater than the set value.

4) The constant scan facility may not operate normally if the scan time becomes temporarily greater than the set value, since the constant scan function is processed by the CPU timers. The set value must therefore be specified with the maximum program scan time fully taken into consideration. The constant scan setting is 40ms in the following example.



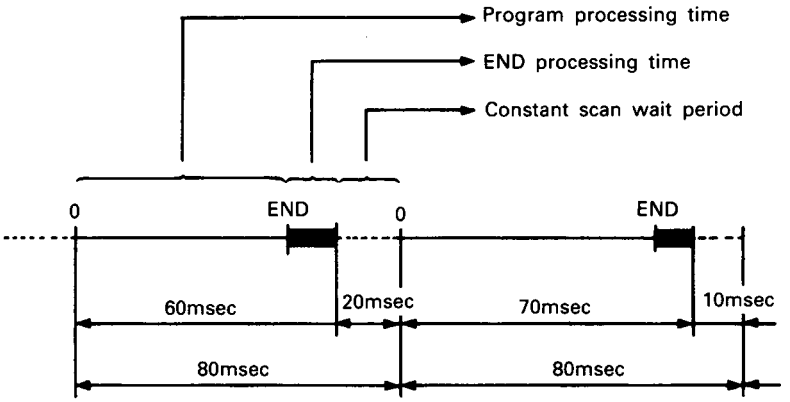
5) The sequence program is not processed after the [END] instruction until the next scan is started. (All device memory data remains as it was before the [END] instruction was processed.) An interrupt program, however, will be executed if its start factor occurs during the wait period prior to the next scan.

6) D9020 data is read after the [END] instruction is executed. When the [CHG] instruction is used, D9020 data is read after the [CHG] instruction is executed.

(3) Operation comparison between direct and refresh modes

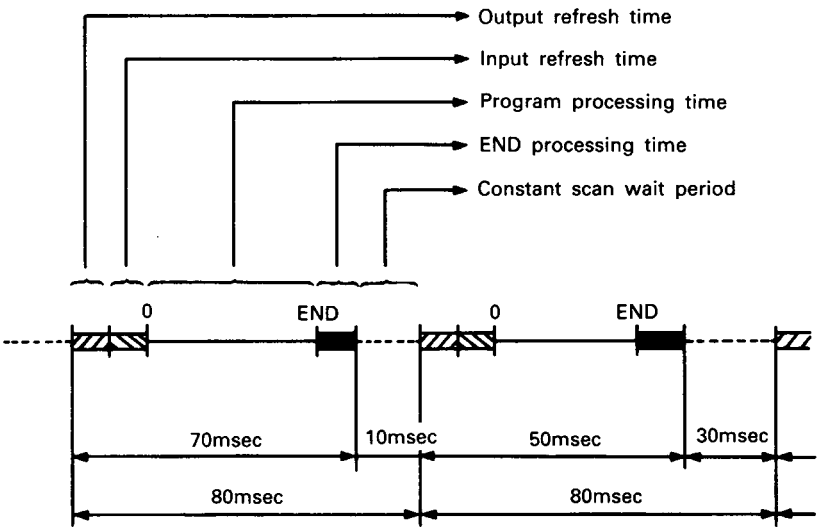
1) Input and output in direct mode

Operation is as shown below when constant scan time is set to 80ms.



2) Input and output in refresh mode

Operation is as shown below when constant scan time is set to 80ms.



(4) Note on setting constant scan time

The constant scan time must be set within the WDT set value. Otherwise a WDT error will occur.

3.6.5 Watch dog timer (WDT)

1) PC internal timer used to detect any PC hardware fault and program error. Defaults to 200ms.

2) The PC resets the WDT to 0 before step 0 is executed (after END processing is executed).

3) The WDT does not time out while the PC operates correctly and executes the [END] (FEND) instruction within the set value in the sequence program.

4) The WDT error occurs when the scan time exceeds the WDT set value and:

   - The PC switches all outputs off and continues operation up to the [END] instruction.
   - 22 is stored to D9008 in BIN on execution of the [END] instruction.
   - M9008 is switched on and 25 is stored to D9008 in BIN.
   - 22 is stored to D9008 in BIN on execution of the [END] instruction.

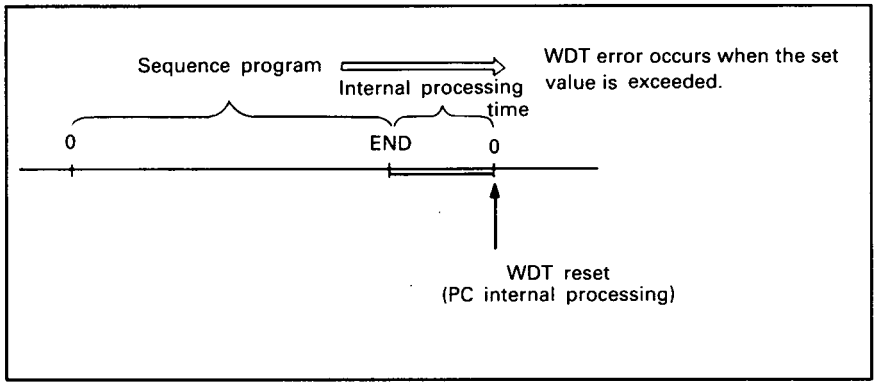


Fig. 3.19 Resetting the Watch Dog Timer

REMARKS

The WDT set value can be changed by parameter setting. (For the parameter setting, see Section 3.5.11.)

3.6.6 Clearing data

Data may either be latched and unlatched in the PC CPU.

(1) Latched data

Latched data remains unchanged if the power is switched off. Latched devices default to L1000-L2047. Other devices and ranges may be set in parameters as required.

1) Latched data can be cleared by setting the RESET switch to LATCH CLEAR or resetting in the program.

2) File registers should be cleared in the program.

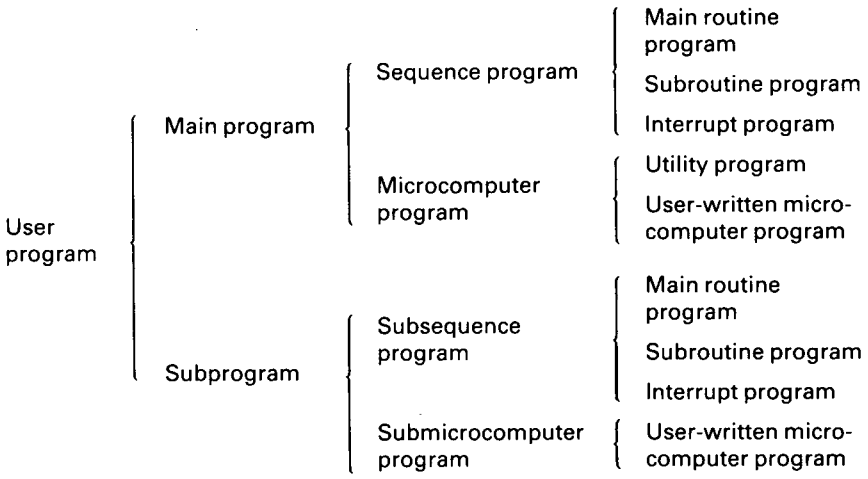
(2) Unlatched data

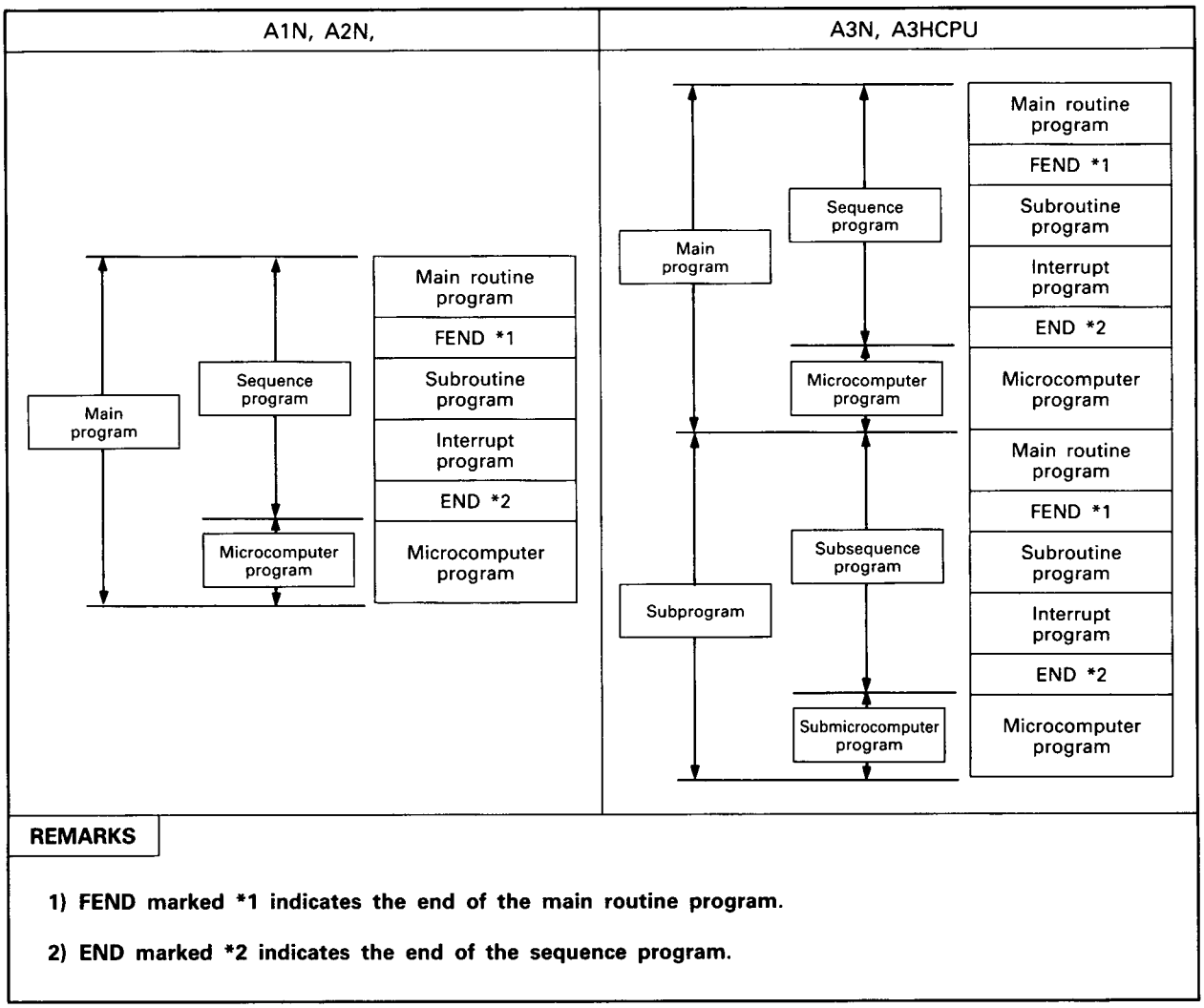
Unlatched data is cleared when the power is switched off.

1) Unlatched devices are cleared when the power is switched from off to on or the RESET switch set to RESET.

3.7 Program Types and Configurations

The following types of program are available on the MELSEC-A series PCs.





REMARKS

1) FEND marked *1 indicates the end of the main routine program.

2) END marked *2 indicates the end of the sequence program.

Fig. 3.20 Program Types and Configurations

3.7.1 Main program

Consists of the sequence program (which is always executed after PC operation is started) and microcomputer program (which is called by the SUB instruction from the sequence program).

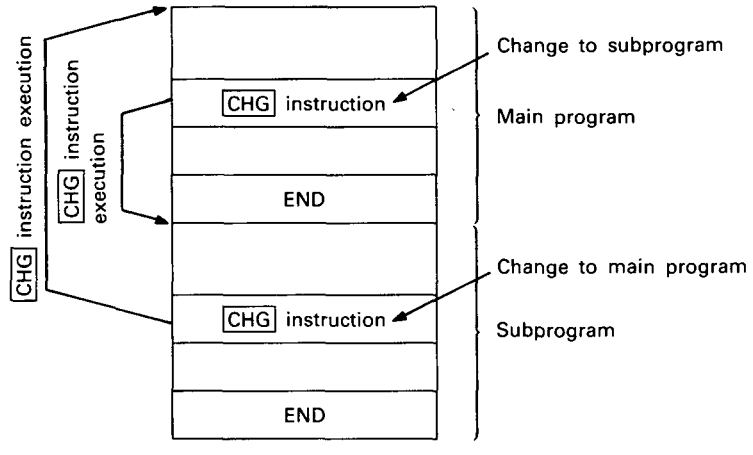
3.7.2 Subprogram

1) The A3N, A3HCPU allows a subprogram to be stored separately from the main program.

2) Like the main program, the subprogram consists of the sequence program and microcomputer program.

3) The main program and subprogram can be executed alternately in series or either program can be separately executed.

4) The CHG instruction is used to switch between the main and subprograms. After entering the main or subprogram, that program is processed repeatedly until the CHG instruction is executed again.



When the CHG instruction is executed, the operating system operates the END processing routine (renewal of T/C present value, self-diagnosis, WDT reset) before the program pointer jumps to the other program area.

Fig. 3.21 Switching between Programs

3.7.3 Sequence program

Consists of the main routine program (which is normally executed) the subroutine program (which is called by the CALL instruction) and the interrupt program (which is called by an interrupt signal).

3.7.4 Microcomputer program

Written in machine language and called by the SUB instruction from the sequence program.

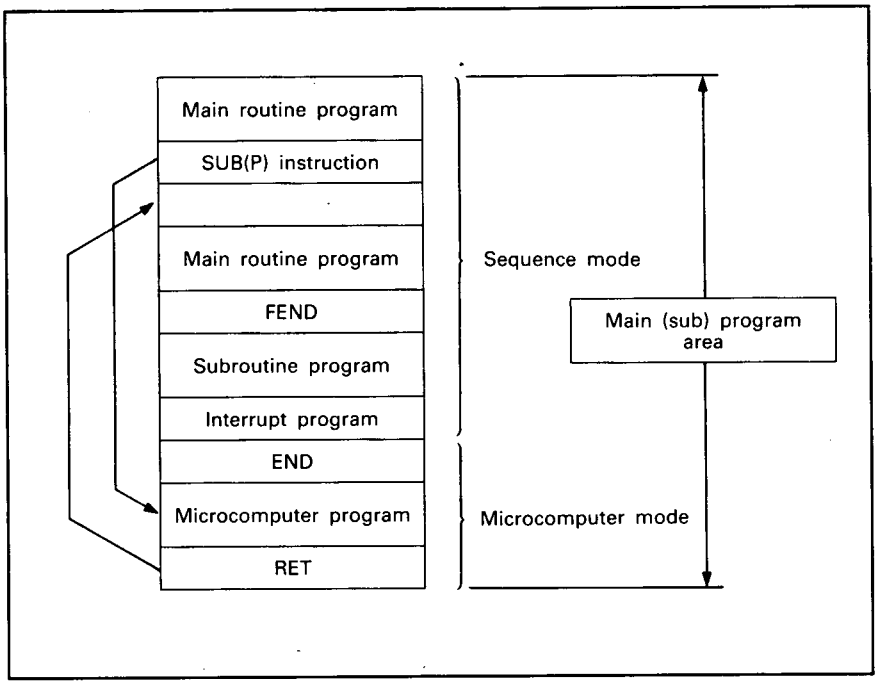


Fig. 3.22 Calling a Microcomputer Program

3.7.5 Main routine program

1) Placed at the head of the sequence program area and executed every scan.
2) Any program in the ACPU cannot be executed without the main routine program.

3.7.6 Subroutine program

1) Executed by the CALL instruction from the main routine program.
2) Used to call a common program routine several times during a scan or when a given condition is enabled.
3) Must be written after the main routine program (after FEND).
4) When the input condition for the CALL instruction is on, the subroutine program is run. When it is off, the main routine program is run.

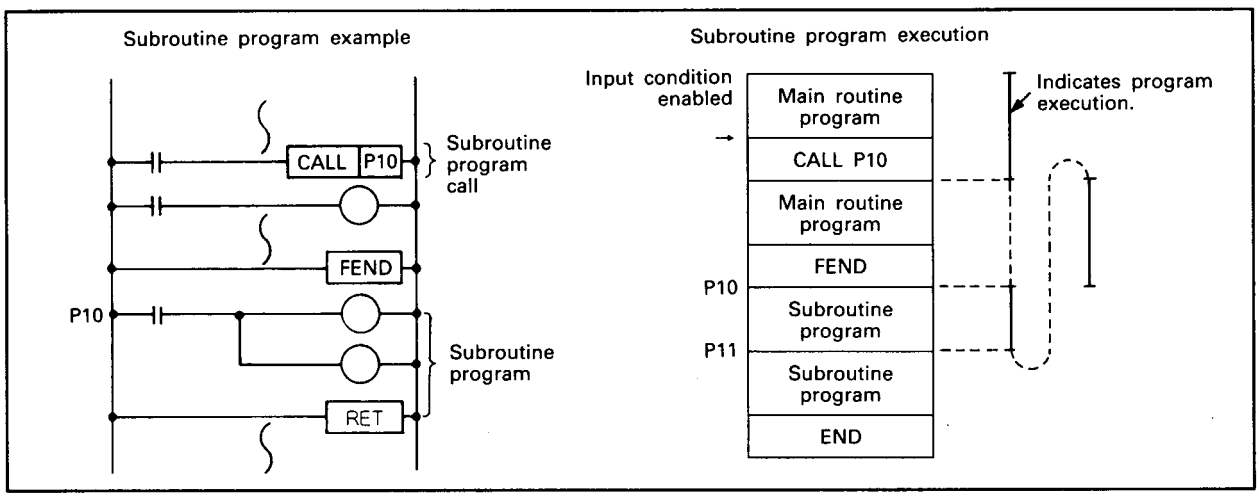


Fig. 3.23 Subroutine Program Execution

Up to five subroutine nesting levels are possible.

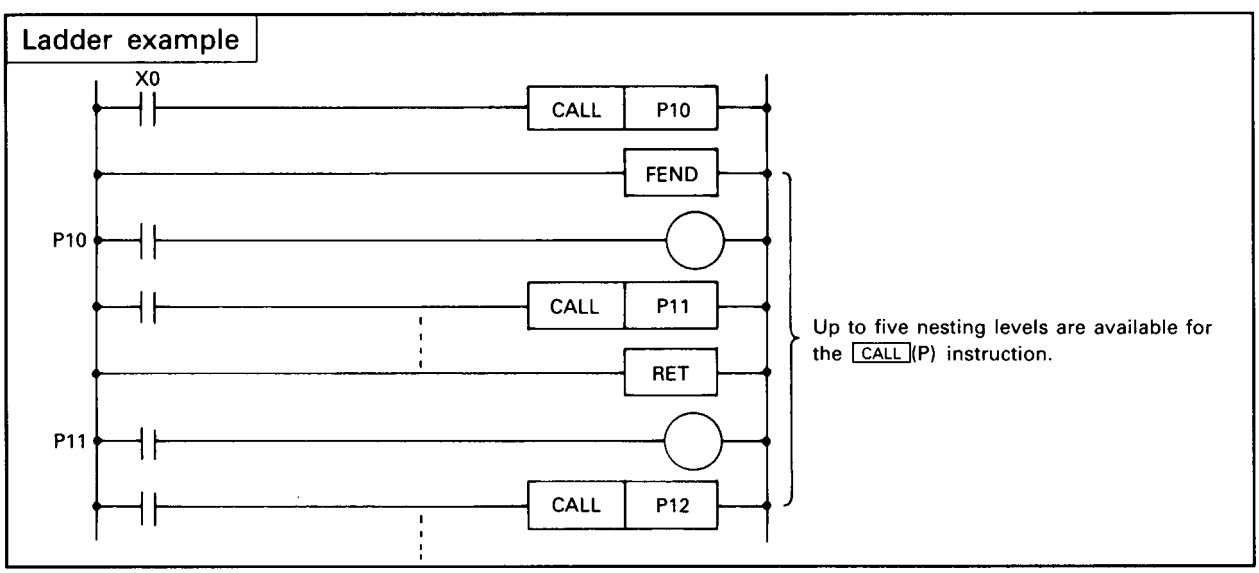


Fig. 3.24 Subroutine Program Example

3.7.7 Interrupt program

Executed when its interrupt factor occurs and written to any of the interrupt pointers I0 to I31.

(1) Interrupt pointer (I)

Interrupt pointers (I) are assigned to interrupt factors as shown in Table 3.3.

Interrupt Pointer	Interrupt Factor		Interrupt Pointer	Interrupt Factor	
I0	From AI61 interrupt input module	1st point	I16	(*1) Interrupt generated by special function module	1st module
I1		2nd point	I17		2nd module
I2		3rd point	I18		3rd module
I3		4th point	I19		4th module
I4		5th point	I20		5th module
I5		6th point	I21		6th module
I6		7th point	I22		7th module
I7		8th point	I23		8th module
I8		9th point	I24	Unused	
I9		10th point	I25		
I10		11th point	I26		
I11		12th point	I27		
I12		13th point	I28		
I13		14th point	I29	Interrupt factor every 40ms of internal timer (*2)	
I14		15th point	I30	Interrupt factor every 20ms of internal timer (*2)	
I15		16th point	I31	Interrupt factor every 10ms of internal timer (*2)	

Table 3.3 Interrupt Pointers and Interrupt Factors

1) *1: Interrupt pointers I16 to I23 are dedicated to interrupt signals generated by special function modules (not AI61). Pointers are assigned to modules in order of I/O allocation.

2) *2: Interrupt pointers I29 to I31 are time based interrupts at intervals of 40ms, 20ms and 10ms in which the interrupt program or count is executed at the specified intervals.

3) The interrupt priority is as follows:

High → I16 to I23, I0 to I15, I31, I30, I29 → Low
Priority

POINT

(1) The [EI] instruction should be written in the program to enable interrupt processing. Any interrupt factor occurring prior to the [EI] instruction will wait to be processed until the [EI] instruction is processed. Any interrupt occurring while the PC is in STOP mode will wait to be processed until after the PC is in RUN mode and the [EI] instruction has been processed.

(2) Any interrupt program may be run during the execution of basic or application instructions.

(2) Interrupt program

The interrupt program is only executed when its interrupt factor occurs and is written to any of the interrupt pointers I0 to I31.

1) The interrupt program must be written after the [FEND] and before the [END] instructions.

2) The interrupt pointer (I) must be written at the beginning of the interrupt program.

3) The [IRET] instruction must be written at the end of each interrupt program in order to return to the sequence program location from which the jump was made to the interrupt program.

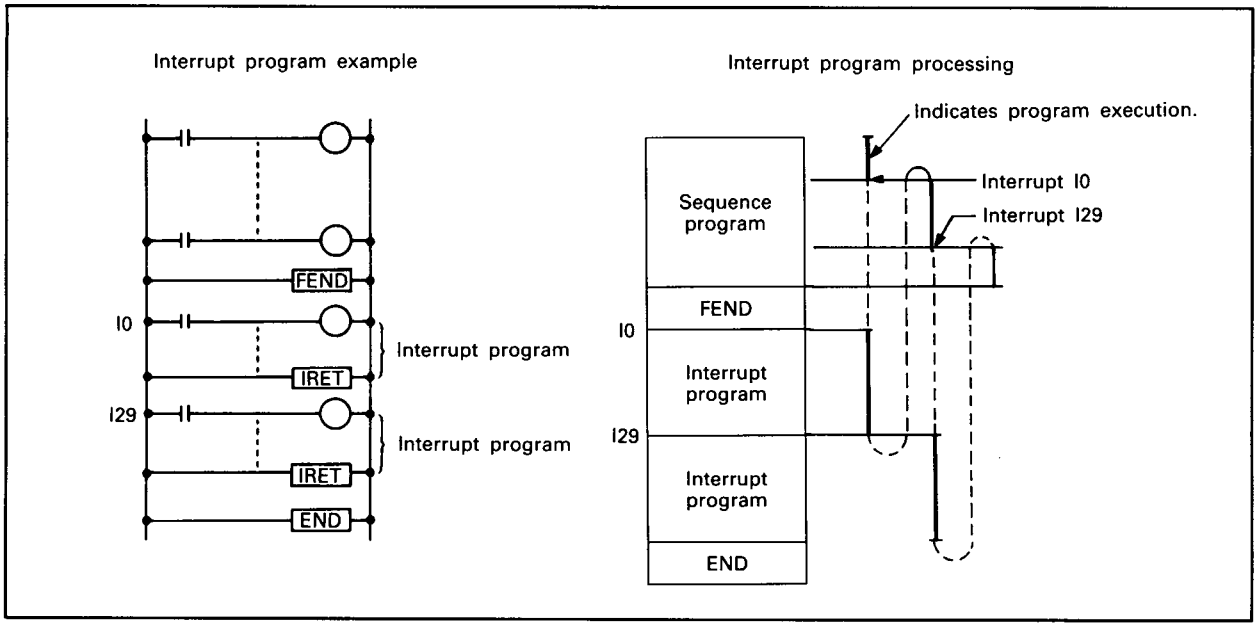


Fig. 3.25 Interrupt Program Processing

3.7.8 Utility program

1) This maker-supplied microcomputer program allows various controls and operations (e.g. PIP control, trigonometric function operation, code conversion) and is called by the SUB instruction from the sequence program.

2) For details of functions, see the corresponding program operating manual.

3) The utility program cannot be stored in the subprogram area.

3.7.9 User-written microcomputer program

1) Written by the user in machine language and called by the SUB instruction from the sequence program.

2) For the writing and storing procedures, see Section 8.

3.8 Using Subprogram

The main/subprogram switching method differs between the A3N and A3HCPUs as described below:

1) In the A3NCPU, the CHG instruction is only executed on the leading edge of its input condition.

2) In the A3HCPU, the CHG instruction is executed repeatedly while its input condition is on.

3) The ACPUs have operation result memories which store operation results during one scan.
The A3N and A3HCPUs have memory areas to store the main and subprograms separately.

3.8.1 Using the CHG instruction with the A3NCPU

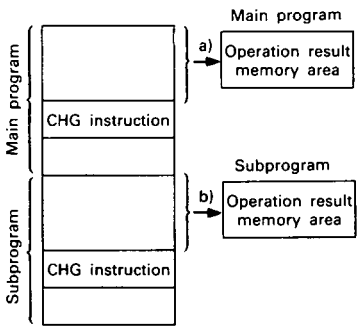
a) The main program is run with the repeatedly triggered instruction executed in accordance with its input condition and the edge triggered instruction executed in accordance with its input condition and previous operation result.

↓ CHG instruction executed

b) The subprogram is run with the repeatedly triggered instruction executed in accordance with its input condition and the edge triggered instruction executed in accordance with its input condition and previous operation result.

↓ CHG instruction executed

Return to step a).



3.8.2 Using the CHG instruction with the A3HCPU

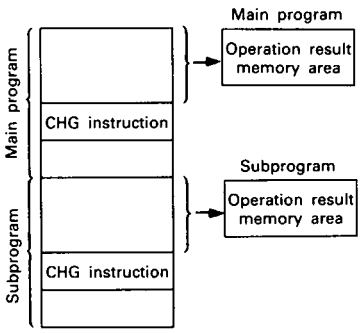
a) The main program is run with the repeatedly triggered instruction executed in accordance with its input condition and the edge triggered instruction executed in accordance with its input condition and previous operation result.

↓ CHG instruction executed

b) The subprogram is run with the repeatedly triggered instruction executed in accordance with its input condition and the edge triggered instruction executed in accordance with its input condition and previous operation result.

↓ CHG instruction executed

Return to step a).



3.8.3 Notes on write during run

1) Programs may be written to a sub or main program area while the PC is running that program area, however, in some cases, control problems may occur (for example, WDT time out). In order to avoid this, it is preferable to write a program into whichever program area, sub or main, is currently not running. In this case, the CHG instruction may be disabled using special relays M9051, M9056 and M9057.

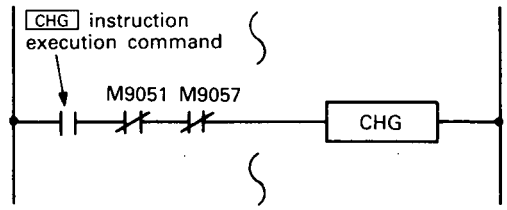
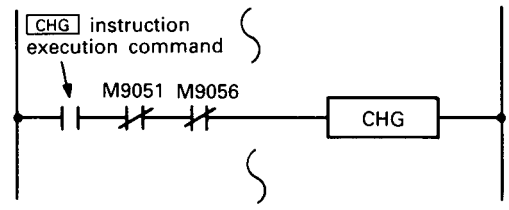
	Main Sequence Program	Subsequence Program
Ladder example		
Operation	The CHG instruction is executed and the main sequence program switched to the subsequence program. The CHG instruction is not executed when M9051 and/or M9057 is on.	The CHG instruction is executed and the subsequence program switched to the main sequence program. The CHG instruction is not executed when M9051 and/or M9056 is on.
Special relays	M9051·····Switched on during main (sub) sequence program transfer to the CPU. Automatically switched off when the transfer is complete. M9056·····Switched on on completion of the main sequence program transfer to the CPU. Automatically switched off on completion of pointer (P) or interrupt pointer (I) address storage. M9057·····Switched on on completion of the subsequence program transfer to the CPU. Automatically switched off on completion of pointer (P) or interrupt pointer (I) address storage.	

Fig. 3.26 Interlocking the CHG Instruction Execution Conditions

2) The main and sub programs are rewritten as shown below.

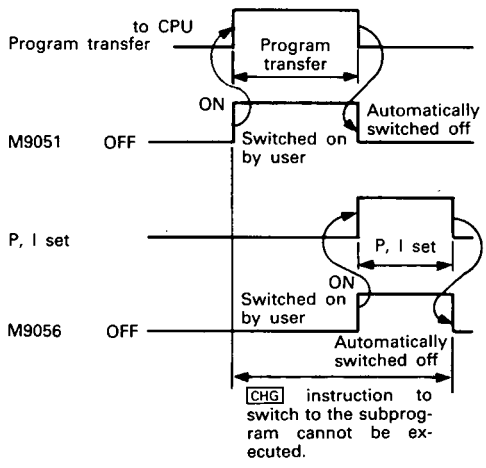
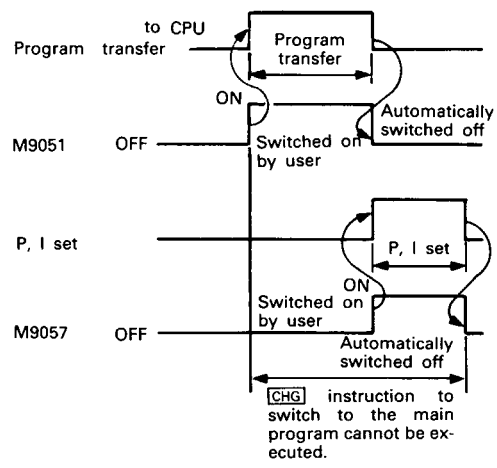
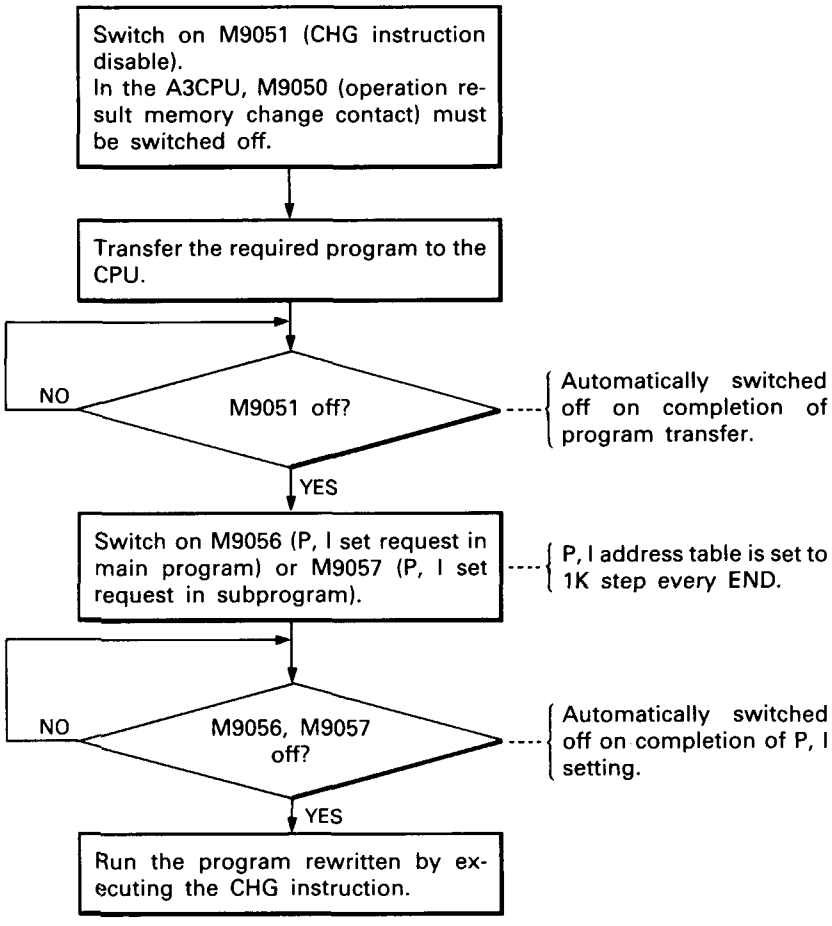
Main Program Rewritten	Subprogram Rewritten
	

Fig. 3.27 Program Rewrite Timing Chart

3) Main (sub) program rewrite procedure



3.8.4 Notes on writing subprogram

1) When there is an interrupt program, the same interrupt programs must be written in the main and subsequence programs using the same interrupt pointer numbers.

2) A timer and a counter cannot be used with the same device number in the main and subsequence programs.
If the timer and counter numbers are the same, the [RST] T/C instruction may be used in the main (sub) sequence program when the [OUT] T/C instruction exists in the (sub) sequence program because the [RST] T/C instruction resets the T/C present value on completion of its execution.

3) Any interrupt is disabled while the [CHG] instruction is being executed. Hence, an interrupt program is only executed after the [CHG] instruction is executed if the corresponding interrupt factor occurs.

4) The pointer (P) indicating the destination of the branch instruction ([CJ], [SCJ], [CALL], [JMP]) may be used with the same numbers in both the main and subsequence programs.

3.9 Using Interrupt Programs

Interrupt status depends on the interrupt condition as described below:

1) Several interrupts have occurred

When several interrupt factors have occurred at the same time, the interrupt programs are executed in accordance with the priority of the corresponding interrupt pointer numbers.

2) Interrupt disable

- If an interrupt occurs during execution of the CHG instruction in the A3N, A3HCPU, the interrupt program is executed after the CHG instruction execution is complete.

- If an interrupt occurs during execution of the FROM / TO instruction, the interrupt program is executed after read/write is complete.

3) Other instructions

The interrupt program is executed during execution of any instruction other than those specified in above 2).

Example:

If an interrupt occurs after 16 bits have been transferred during execution of the 32-bit data transfer (DMOV) instruction, the DMOV instruction is stopped and the interrupt program executed. The remaining 16 bits are transferred after the interrupt program run is complete.
When storing data used in the interrupt program by using the sequence program, the DI, EI instructions should be used to disable the interrupt program until execution of the data storage area is complete.

4) Real time interrupt

Interrupt status of interrupt pointers I29 to I31 (real time interrupt pointers) depends on the CPU used as described below:

A3HCPU·····If link refresh is being executed when an interrupt has occurred, the interrupt waits to be processed until the refresh is complete.

A1N, A2N, A3NCPU·····If link refresh is being executed when an interrupt has occurred, the interrupt program is executed immediately.

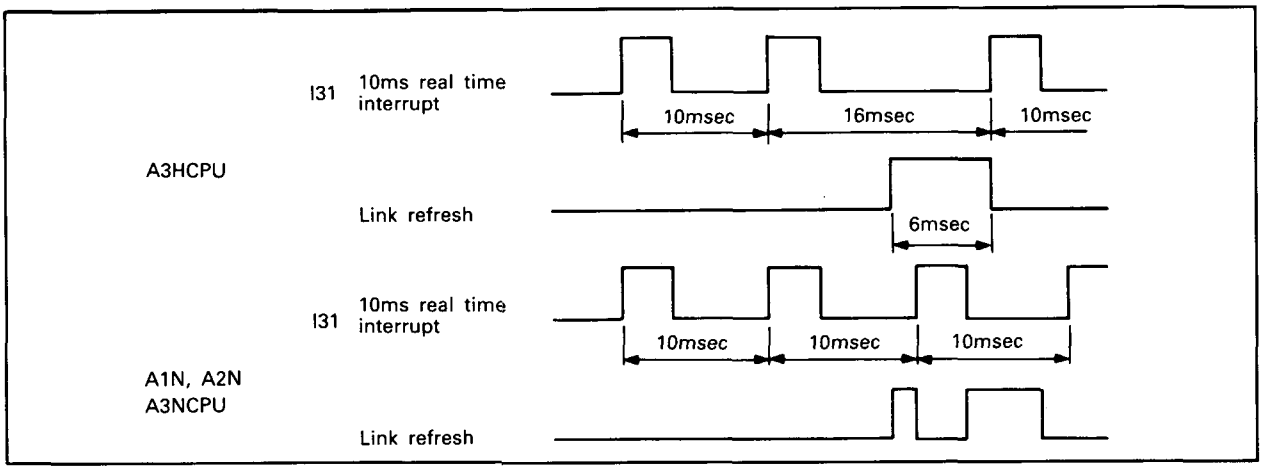


Fig. 3.28 Real Time Interrupt Status

5) Interrupt during [END] processing

If an interrupt occurs during [END] processing, the interrupt waits to be processed until the [END] processing is complete only when the other station status is being monitored in a data link system.

6) Interrupt during constant scan wait period

An interrupt program corresponding to the interrupt factor is executed if the interrupt occurs during the wait period with the constant scan facility set.

3.9.1 Notes on writing interrupt programs

1) Any device switched on by the [PLS] instruction in an interrupt program remains on until that interrupt program is executed.

2) Interrupt is disabled while an interrupt program is being executed. The [EI], [DI] instructions should not be executed in any interrupt program.

3) The status of any device used in the interrupt program remains the same after the interrupt program has been executed.

4) Interrupt during alternate run of main and subprograms

   a) The interrupt program in the currently running program is executed when there are interrupt programs in both the main and subprograms.

   b) If one program has an interrupt program and the other not, the following occurs when an interrupt occurs during run of the program without any interrupt program:

   - I0 to I23·····Results in an error and stops the PC.
   - I29 to I31·····Ignored.

5) Timers must not be used in any interrupt program. Timer contacts may be switched on or the present value may be equal to the set value if the timer coil is off.

3.10 Using Annunciators

(1) A1N, A2NCPU·····The ERROR LED on the CPU front flickers when the annunciator is switched on.

(2) A3N, A3HCPU·····An F number is indicated on the LED display when the annunciator is switched on. The detected F number and its comment can be displayed alternately by parameter setting.

(3) When many annunciators are used, shorter scan time can be achieved by using the [SET] F[ ] instruction than by using the [OUT] F[ ] instruction.
This is because the [SET] F[ ] instruction is only executed on the leading edge of its input condition whereas the [OUT] F[ ] instruction is executed independently of its input condition.

(4) Reset the annunciator coil switched on by the [SET] F[ ] instruction as described below:

1) Annunciator number detected by the [SET] F[ ] instruction

- Execute the [RST] F[ ] instruction.

2) Annunciator number stored in D9009, D9125

- Execute the LEDR instruction.
- Press the INDICATOR RESET switch on the front of the A3N, A3HCPU.

(5) When a comment is required, write it to the F number device using any of the following characters:

- Numeral·····0 to 9
- Alphabet·····A to Z (Capital)
- Special symbol····· <, > , =, ✕, /, ', +, −

(6) Any annunciator switched on by other than the [SET] F[ ] or [OUT] F[ ] instruction is processed in the same way as the internal relay and the functions described in above (1) to (5) are not available.

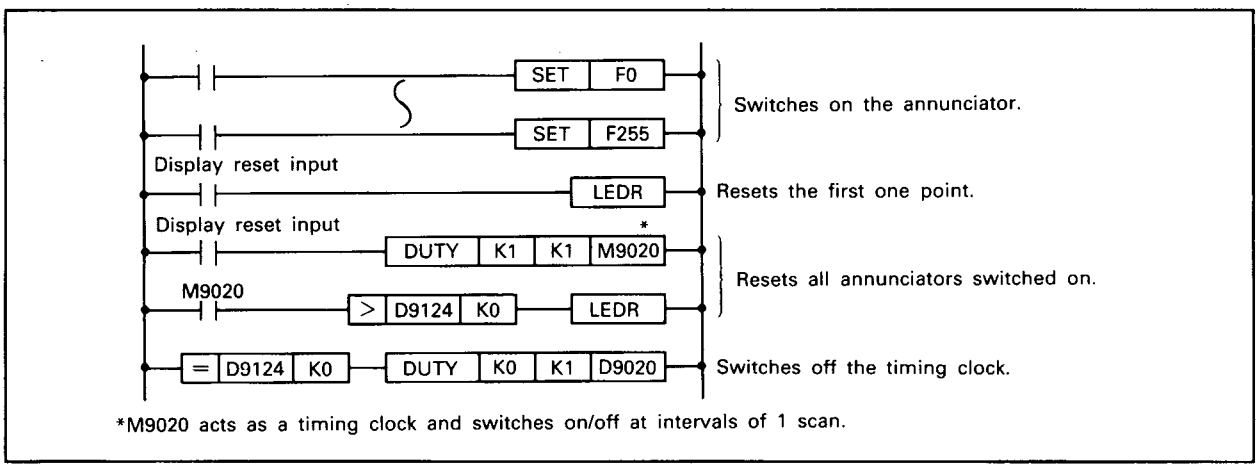
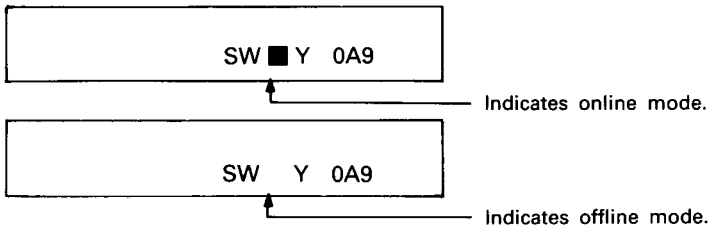


Fig. 3.29 Annunciator Display Program

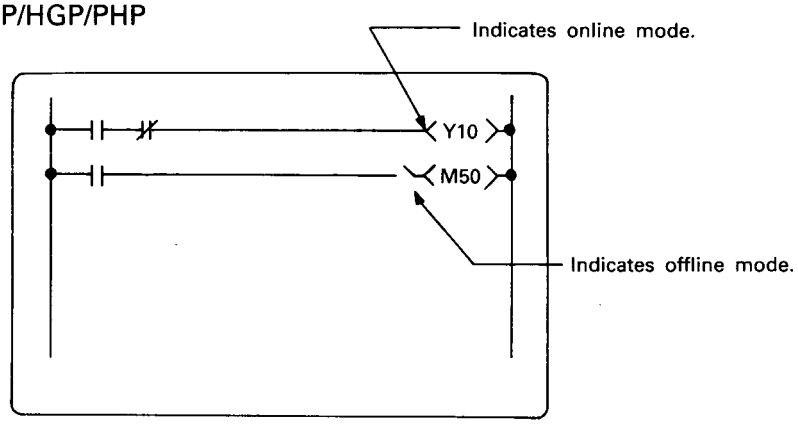
3.11 Offline Switch

Separates Y, M, L, S, B, F coil (OUT instruction) from the sequence program and retains its ON/OFF status independently of the OUT instruction execution condition.
This function is useful for program debugging or during machine adjustment.
The offline switch is used in test mode of the peripheral device.

(1) PU



(2) GPP/HGP/PHP



For further details, see the corresponding peripheral device operating manual.

4. INSTRUCTIONS

4.1 Classification

The instructions of MELSEC-A series are largely classified into sequence instructions, basic instructions, and application instructions. These instructions are shown in Table 4.1.

Classification of instructions		Description	Page
Sequence instruction	Contact instruction	Operation start, series connection, parallel connection	5-2 to 5-4
	Connection instruction	Ladder block connection, operation result storage/read	5-5 to 5-13
	Output instruction	Bit device output, pulse output, output reverse	5-14 to 5-25
	Shift instruction	Bit device shift	5-26 to 5-27
	Master control instruction	Master control	5-28 to 5-29
	Program branch instruction	Program jump, subroutine/interrupt program call	5-30 to 5-40
	Program switching instruction	Switching between main and subprogram	5-41 to 5-48
	FOR to NEXT instruction	Program repeated between FOR and NEXT instruction	5-49 to 5-50
	Refresh instruction	Link refresh, partial refresh execution	5-51 to 5-56
	Termination instruction	Program termination	5-57 to 5-60
	Other instructions	Program stop, no operation, etc.	5-61 to 5-64
Basic instruction	Comparison operation instruction	Comparison such as =, >, and <	6-2 to 6-7
	Arithmetic operation instruction	Addition, subtraction, multiplication, and division of BIN and BCD	6-8 to 6-37
	BCD ⟷ BIN conversion instruction	Conversion from BCD to BIN and BIN to BCD	6-38 to 6-44
	Data transfer instruction	Transfer of specified data	6-45 to 6-55
Application instruction	Logical operation instruction	Logical operation such as logical sum and logical product	7-2 to 7-20
	Rotation instruction	Rotation of specified data	7-21 to 7-29
	Shift instruction	Shift of specified data	7-30 to 7-36
	Data processing instruction	Data processing such as 16-bit data search, decode, and encode	7-37 to 7-52
	FIFO instruction	Read/write of FIFO table	7-53 to 7-57
	Buffer memory access instruction	Special function module and read/write	7-58 to 7-62
	Local, remote I/O station access instruction	Local, remote I/O station data read/write	7-63 to 7-70
	Display instruction	ASCII code print, character display on LED, etc.	7-71 to 7-85
	Others	Instructions which are not included in the above classification, such as WDT reset, and set/reset of carry flag.	7-86 to 7-96

Table 4.1 Classification of Instructions

4.2 Instruction List

4.2.1 Explanation for instructions lists

Instruction lists in Section 4.2.2 to 4.2.4 are in the following format.

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
BIN 16-bit addition/ subtraction	16 bits	+	+ Ⓢ Ⓓ	(D) + (S) → (D)	⎍	5	●	●		●	6-10 to 6-12	●	●	●	●
		+P	+P Ⓢ Ⓓ		⎎	5	●	●		●	6-10 to 6-12	●	●	●	●
		+	+ (S1) (S2) Ⓓ	(S1) + (S2) → (D)	⎍	7	●	●		●	6-10 to 6-12	●	●	●	●
		+P	+P (S1) (S2) Ⓓ		⎎	7	●	●		●	6-10 to 6-12	●	●	●	●
BIN 32-bit addition/ subtraction	32 bits	D+	D+ Ⓢ Ⓓ	(D1, D) + (S+1, S) → (D+1, D)	⎍	9	●	●		●	6-13 to 6-15	●	●	●	●
		D+P	D+P Ⓢ Ⓓ		⎎	9	●	●		●	6-13 to 6-15	●	●	●	●
		D+	D+ (S1) (S2) Ⓓ	(S1+1, S1) + (S2+1, S2) → (D+1, D)	⎍	11	●	●		●	6-13 to 6-15	●	●	●	●
		D+P	D+P (S1) (S2) Ⓓ		⎎	11	●	●		●	6-13 to 6-15	●	●	●	●

① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩ ⑪ ⑫ ⑬

Table 4.2 Explanation for Instructions Lists

Explanation

①·····Classifies the instructions by applications.

②·····Indicates the unit of processing at the execution of instruction.

Unit of Processing	Device	Number of Points
16 bits	X, Y, M, L, S, F, B	Max. 16 points in units of 4 points.
	T, C, D, W, R, A, Z, V	1 point
32 bits	X, Y, M, L, S, F, B	Max. 32 points in units of 4 points.
	T, C, D, W, R, A0, Z	2 points

③·····Indicates the instruction symbol used for the program. The instruction symbol is shown on a 16-bit instruction basis. The symbols of a 32-bit instruction and an instruction executed only at the rise from OFF to ON are as indicated below:

32-bit instruction ····· D is added to the head of instruction.

Example: + ⇨ D+

16-bit instruction　　32-bit instruction

Instruction executed only at the rise from OFF to ON ····· P is added to the end of instruction.

Example: + ⇨ +P

Instruction executed during ON　　Instruction executed only at the rise from OFF to ON

④·····Indicates the symbol diagram in the circuit.

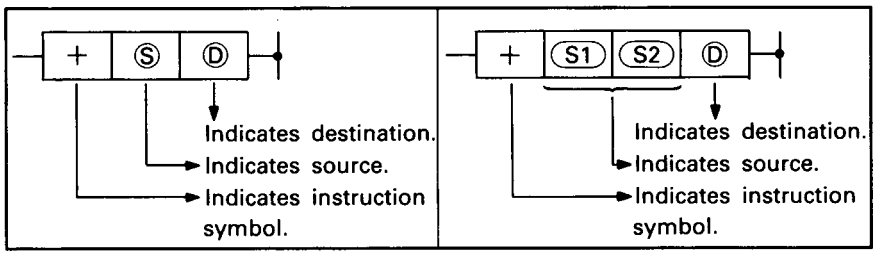
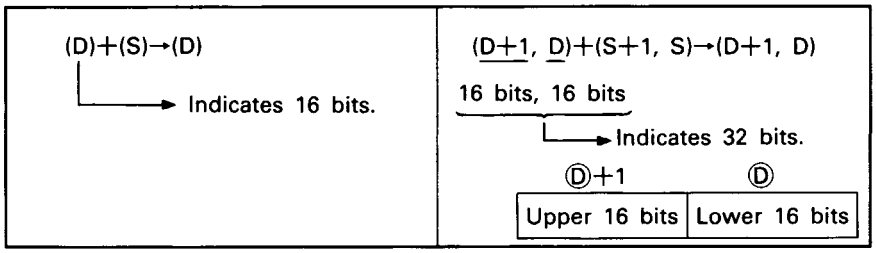


Fig. 4.2 Symbol Representations in Ladder

Destination: Indicates the destination of data after operation.
Source: Stores data before operation.

⑤·····Indicates the processing of each instruction.



⑥·····Indicates the execution condition of each instruction and details are as described below:

Symbol	Execution Condition
No entry	Instruction which is always executed.
	Instruction which is executed during ON. Executes instruction only while the preceding condition of that instruction is on. When the preceding condition is off, that instruction is not executed and not processed.
	Instruction which is executed once during ON. Executes instruction only at the positive transition of the preceding condition of instruction, i.e. the condition changes from off to on. Thereafter, even if the condition is on, that instruction is not executed and not processed.
	Instruction which is executed once during OFF. Executes instruction only at the negative transition of the preceding condition of instruction, i.e. the condition changes from on to off. Thereafter, even is the condition is off, that instruction is not executed and not processed.

⑦·····Indicates the number of steps of each instruction. The number of steps, which change depending on conditions, is indicated in two stages. For details, refer to each instruction.

⑧·····The ● mark indicates that the instruction can be indexed (Z, V).

⑨·····The ● mark indicates that the instruction is a subset instruction.

⑩·····The ● mark indicates that a carry flag will change.

⑪·····The ● mark indicates that an error flag will turn on at operation error time.

⑫·····Indicates a page which explains each instruction.

⑬·····The ● mark indicates that the instruction can be used.

IB (NA) 66147-A

4.2.2 Sequence instructions

(1) Contact instructions

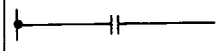
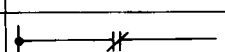
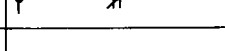
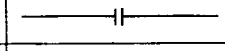
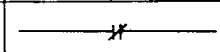
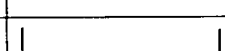
Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Contact	—	LD		Logical operation start (NO contact operation start)		1					5-2 to 5-4	●	●	●	●
		LDI		Logical NOT operation start (NC contact operation start)		1						●	●	●	●
		AND		Logical product (NO contact series connection)		1						●	●	●	●
		ANI		Logical product NOT (NC contact series connection)		1						●	●	●	●
		OR		Logical add (NO contact parallel connection)		1						●	●	●	●
		ORI		Logical add NOT (NC contact parallel connection)		1						●	●	●	●

Table 4.3 Contact Instructions

(2) Connection instructions

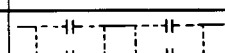
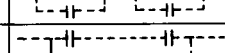
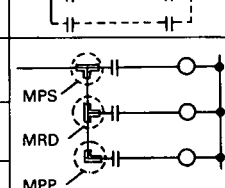
Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Connection	—	ANB		ANDs logical blocks. (Series connection of blocks)		1					5-5 to 5-8	●	●	●	●
		ORB		ORs logical blocks. (Parallel connection of blocks)		1						●	●	●	●
		MPS		Stores the operation result.		1					5-9 to 5-13	●	●	●	●
		MRD		Reads the operation result from MPS.		1						●	●	●	●
		MPP		Reads the operation result from MPS and clears the result.		1						●	●	●	●

Table 4.4 Connection Instructions

(3) Output instructions

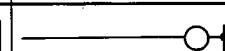
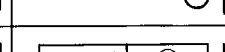
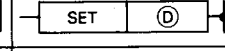
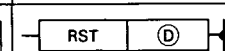
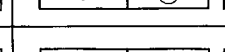
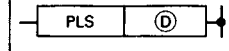
Classification		Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
OUT	—	OUT		Device output		1 3					5-14 to 5-17	●	●	●	●
		SET		Device set	*	1 3					5-18 to 5-21	●	●	●	●
		RST		Device reset	*	1 3						●	●	●	●
		PLS		Generates one-program cycle pulses on the leading edge of input signal.		3					5-22 to 5-23	●	●	●	●
		PLF		Generates one-program cycle pulses on the trailing edge of input signal.		3						●	●	●	●
		CHK		Device output reverse Valid in I/O refresh mode		5					5-24 to 5-25	●	●	●	—

Table 4.5 Output Instructions

(4) Shift instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Shift	—	SFT	SFT D	Shifts device 1 bit	⎍	3					5-26 to 5-27	●	●	●	●
		SFTP	SFTP D		⎎	3						●	●	●	●

Table 4.6 Shift Instructions

(5) Master control instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Master control	—	MC	MC n D	Master control start		5					5-28 to 5-29	●	●	●	●
		MCR	MCR n	Master control reset		3						●	●	●	●

Table 4.7 Master Control Instructions

(6) Program branch instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Jump	—	CJ	CJ P✱✱	Jumps to P✱✱ after the input condition is enabled.	⎍	3	●			●	5-30 to 5-33	●	●	●	●
		SCJ	SCJ P✱✱	Jumps to P✱✱ beginning with the next scan after the input condition is enabled.		3	●			●		●	●	●	●
		JMP	JMP P✱✱	Unconditionally jumps to P✱✱.		3	●			●		●	●	●	●
Subroutine call	—	CALL	CALL P✱✱	Executes the subroutine program at P✱✱ after the input condition is enabled.	⎍	3	●			●	5-34 to 5-35	●	●	●	●
		CALLP	CALLP P✱✱		⎎	3	●			●		●	●	●	●
		RET	RET	Returns execution from the subroutine program to the sequence program.		1						●	●	●	●
Interrupt program call	—	EI	EI	Enables interrupt program run. Valid for A□N with M9053 off.		1					5-36 to 5-38	●	●	●	●
		DI	DI	Disables interrupt program run. Valid for A□N with M9053 off.		1						●	●	●	●
		IRET	IRET	Returns execution from the interrupt program to the sequence program.		1						●	●	●	●
Microcomputer program call	—	SUB	SUB n	Executes the microcomputer program specified by n.	⎍	3	●			●	5-39 to 5-40	●	●	●	●
		SUBP	SUBP n		⎎	3	●			●		●	●	●	●

Table 4.8 Program Branch Instructions

(7) Program switching instruction

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Switching	—	CHG	CHG	Switches between the main and subprograms.	A3 A3N ⎎ A3H ⎍	1					5-41 to 5-48	—	—	●	●

Table 4.9 Program Switching Instruction

(8) FOR / NEXT instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Repetition	—	FOR	FOR n	Executes the program area between FOR and NEXT "n" times.		3				●	5-49 to 5-50	●	●	●	●
		NEXT	NEXT			1				●	5-49 to 5-50	●	●	●	●

Table 4.10 FOR / NEXT Instructions

(9) Refresh instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Link refresh	—	COM	COM	Executes link refresh, general data processing.	⊓	3					5-51 to 5-52	●	●	●	●
Link refresh enable, disable	—	EI	EI	Enables link refresh. Valid when M9053 is on.		1					5-53 to 5-54	●	●	●	—
		DI	DI	Disables link refresh. Valid when M9053 is on.		1						●	●	●	—
Partial refresh	—	SEG	SEG Ⓢ n	Only executes refresh for the corresponding device during 1 scan. Valid when M9052 is on.	⊓	7					5-55 to 5-56	●	●	●	●

Table 4.11 Refresh Instructions

(10) Termination instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Program end	—	FEND	FEND	Always used at the end of the main routine program to terminate processing.		1				●	5-57 to 5-58	●	●	●	●
	—	END	—	Always used at the end of the sequence program to return to step 0.		1					5-59 to 5-60	●	●	●	●

Table 4.12 Termination Instructions

(11) Other instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Stop	—	STOP	STOP	Resets output after the input condition is enabled, and stops the sequence program. The sequence program is resumed by setting the RUN key switch to RUN.	⊓	1					5-61 to 5-62	●	●	●	●
No operation	—	NOP	—	No operation For program erasure or space		1					5-63 to 5-64	●	●	●	●

Table 4.13 Other Instructions

REMARKS

Execution Condition marked * in (3) Output instructions:

when the device used is F (annunciator).

when the other device is used.

4.2.3 Basic instructions

(1) Comparison instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
16-bit data comparison	16 bits	LD=	LD= (S1) (S2)	Continuity when (S1) = (S2) Non-continuity when (S1) ≠ (S2)		5 7	●	●		●	6-4 to 6-5	●	●	●	●
		AND=	AND= (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		OR=	OR= (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		LD<>	LD<> (S1) (S2)	Continuity when (S1) ≠ (S2) Non-continuity when (S1) = (S2)		5 7	●	●		●	6-4 to 6-5	●	●	●	●
		AND<>	AND<> (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		OR<>	OR<> (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		LD>	LD> (S1) (S2)	Continuity when (S1) > (S2) Non-continuity when (S1) ≦ (S2)		5 7	●	●		●	6-4 to 6-5	●	●	●	●
		AND>	AND> (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		OR>	OR> (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		LD<=	LD<= (S1) (S2)	Continuity when (S1) ≦ (S2) Non-continuity when (S1) > (S2)		5 7	●	●		●	6-4 to 6-5	●	●	●	●
		AND<=	AND<= (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		OR<=	OR<= (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		LD<	< (S1) (S2)	Continuity when (S1) < (S2) Non-continuity when (S1) ≧ (S2)		5 7	●	●		●	6-4 to 6-5	●	●	●	●
		AND<	< (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		OR<	< (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		LD>=	>= (S1) (S2)	Continuity when (S1) ≧ (S2) Non-continuity when (S1) < (S2)		5 7	●	●		●	6-4 to 6-5	●	●	●	●
		AND>=	>= (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
		OR>=	>= (S1) (S2)			5 7	●	●		●	6-4 to 6-5	●	●	●	●
32-bit data comparison	32 bits	LDD=	LDD= (S1) (S2)	Continuity when (S1+1, S1) = (S2+1, S2) Non-continuity when (S1+1, S1) ≠ (S2+1, S2)		11	●			●	6-6 to 6-7	●	●	●	●
		ANDD=	ANDD= (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		ORD=	ORD= (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		LDD<>	LDD<> (S1) (S2)	Continuity when (S1+1, S1) ≠ (S2+1, S2) Non-continuity when (S1+1, S1) = (S2+1, S2)		11	●			●	6-6 to 6-7	●	●	●	●
		ANDD<>	ANDD<> (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		ORD<>	ORD<> (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●

Table 4.14 Comparison Operation Instructions (Continue)

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
32-bit data comparison	32 bits	LDD>	LDD> (S1) (S2)	Continuity when (S1+1, S1) > (S2+1, S2) Non-continuity when (S1+1, S1) ≦ (S2+1, S2)		11	●			●	6-6 to 6-7	●	●	●	●
		ANDD>	ANDD> (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		ORD>	ORD> (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		LDD<=	LDD<= (S1) (S2)	Continuity when (S1+1, S1) ≦ (S2+1, S2) Non-continuity when (S1+1, S1) > (S2+1, S2)		11	●			●	6-6 to 6-7	●	●	●	●
		ANDD<=	ANDD<= (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		ORD<=	ORD<= (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		LDD<	D< (S1) (S2)	Continuity when (S1+1, S1) < (S2+1, S2) Non-continuity when (S1+1, S1) ≧ (S2+1, S2)		11	●			●	6-6 to 6-7	●	●	●	●
		ANDD<	D< (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		ORD<	D< (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		LDD>=	D>= (S1) (S2)	Continuity when (S1+1, S1) ≧ (S2+1, S2) Non-continuity when (S1+1, S1) < (S2+1, S2)		11	●			●	6-6 to 6-7	●	●	●	●
		ANDD>=	D>= (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●
		ORD>=	D>= (S1) (S2)			11	●			●	6-6 to 6-7	●	●	●	●

Table 4.14 Comparison Operation Instructions

(2) Arithmetic operation instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
BIN 16-bit addition/subtraction	16 bits	+	+ (S) (D)	(D) + (S) → (D)		5	●	●		●	6-10 to 6-12	●	●	●	●
		+P	+P (S) (D)			5	●	●		●	6-10 to 6-12	●	●	●	●
		+	+ (S1) (S2) (D)	(S1) + (S2) → (D)		7	●	●		●	6-10 to 6-12	●	●	●	●
		+P	+P (S1) (S2) (D)			7	●	●		●	6-10 to 6-12	●	●	●	●
		−	− (S) (D)	(D) − (S) → (D)		5	●	●		●	6-10 to 6-12	●	●	●	●
		−P	−P (S) (D)			5	●	●		●	6-10 to 6-12	●	●	●	●
		−	− (S1) (S2) (D)	(S1) − (S2) → (D)		7	●	●		●	6-10 to 6-12	●	●	●	●
		−P	−P (S1) (S2) (D)			7	●	●		●	6-10 to 6-12	●	●	●	●
BIN 16-bit addition/subtraction	32 bits	D+	D+ (S) (D)	(D + 1, D) + (S + 1, S) → (D + 1, D)		9	●	●		●	6-13 to 6-15	●	●	●	●
		D+P	D+P (S) (D)			9	●	●		●	6-13 to 6-15	●	●	●	●
		D+	D+ (S1) (S2) (D)	(S1 + 1, S1) + (S2 + 1, S2) → (D + 1, D)		11	●	●		●	6-13 to 6-15	●	●	●	●
		D+P	D+P (S1) (S2) (D)			11	●	●		●	6-13 to 6-15	●	●	●	●

Table 4.15 Arithmetic Operation Instructions (Continue)

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
BIN 32-bit addition/ subtraction	32 bits	D−	D− Ⓢ Ⓓ	(D + 1, D) − (S + 1, S) → (D + 1, D)	⌐¬	9	●	●		●	6-13 to 6-15	●	●	●	●
		D−P	D−P Ⓢ Ⓓ		⌐↑	9	●	●		●	6-13 to 6-15	●	●	●	●
		D−	D− (S1) (S2) Ⓓ	(S1 + 1, S1) − (S2 + 1, S2) → (D + 1, D)	⌐¬	11	●	●		●	6-13 to 6-15	●	●	●	●
		D−P	D−P (S1) (S2) Ⓓ		⌐↑	11	●	●		●	6-13 to 6-15	●	●	●	●
BIN 16-bit addition/ subtraction	16 bits	✳	✳ (S1) (S2) Ⓓ	(S1) × (S2) → (D + 1, D)	⌐¬	7	●	●		●	6-16 to 6-18	●	●	●	●
		✳P	✳P (S1) (S2) Ⓓ		⌐↑	7	●	●		●	6-16 to 6-18	●	●	●	●
		／	／ (S1) (S2) Ⓓ	(S1) ÷ (S2) → Quotient (D), Remainder (D + 1)	⌐¬	7	●	●		●	6-19 to 6-21	●	●	●	●
		／P	／P (S1) (S2) Ⓓ		⌐↑	7	●	●		●	6-19 to 6-21	●	●	●	●
BIN 32-bit addition/ subtraction	32 bits	D✳	D✳ (S1) (S2) Ⓓ	(S1 + 1, S1) × (S2 + 1, S2) → (D + 3, D + 2, D + 1, D)	⌐¬	11	●	●		●	6-16 to 6-18	●	●	●	●
		D✳P	D✳P (S1) (S2) Ⓓ		⌐↑	11	●	●		●	6-16 to 6-18	●	●	●	●
		D／	D／ (S1) (S2) Ⓓ	(S1 + 1, S1) ÷ (S2 + 1, S2) → Quotient (D + 1, D), Remainder (D + 3, D + 2)	⌐¬	9	●	●		●	6-19 to 6-21	●	●	●	●
		D／P	D／P (S1) (S2) Ⓓ		⌐↑	9	●	●		●	6-19 to 6-21	●	●	●	●
BCD 4-digit addition, subtraction	BCD 4-digits	B+	B+ Ⓢ Ⓓ	(D) + (S) → (D)	⌐¬	7	●			●	6-22 to 6-24	●	●	●	●
		B+P	B+P Ⓢ Ⓓ		⌐↑	7	●			●	6-22 to 6-24	●	●	●	●
		B+	B+ (S1) (S2) Ⓓ	(S1) + (S2) → (D)	⌐¬	9	●			●	6-22 to 6-24	●	●	●	●
		B+P	B+P (S1) (S2) Ⓓ		⌐↑	9	●			●	6-22 to 6-24	●	●	●	●
		B−	B− Ⓢ Ⓓ	(D) − (S) → (D)	⌐¬	7	●			●	6-22 to 6-24	●	●	●	●
		B−P	B−P Ⓢ Ⓓ		⌐↑	7	●			●	6-22 to 6-24	●	●	●	●
		B−	B− (S1) (S2) Ⓓ	(S1) − (S2) → (D)	⌐¬	9	●			●	6-22 to 6-24	●	●	●	●
		B−P	B−P (S1) (S2) Ⓓ		⌐↑	9	●			●	6-22 to 6-24	●	●	●	●
BCD 8-digit addition, subtraction	BCD 8-digits	DB+	DB+ Ⓢ Ⓓ	(D + 1, D) + (S + 1, S) → (D + 1, D)	⌐¬	9	●			●	6-25 to 6-27	●	●	●	●
		DB+P	DB+P Ⓢ Ⓓ		⌐↑	9	●			●	6-25 to 6-27	●	●	●	●
		DB+	DB+ (S1) (S2) Ⓓ	(S1 + 1, S1) + (S2 + 1, S2) → (D + 1, D)	⌐¬	11	●			●	6-25 to 6-27	●	●	●	●
		DB+P	DB+P (S1) (S2) Ⓓ		⌐↑	11	●			●	6-25 to 6-27	●	●	●	●

Table 4.15 Arithmetic Operation Instructions (Continue)

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
BCD 8-digit addition, subtraction	BCD 8-digits	DB−	DB− S D	(D + 1, D) − (S + 1, S) → (D + 1, D)	⌐¬	9	●			●	6-25 to 6-27	●	●	●	●
		DB−P	DB−P S D		⌐↑	9	●			●	6-25 to 6-27	●	●	●	●
		DB−	DB− S1 S2 D	(S1 + 1, S1) − (S2 + 1, S2) → (D + 1, D)	⌐¬	11	●			●	6-25 to 6-27	●	●	●	●
		DB−P	DB−P S1 S2 D		⌐↑	11	●			●	6-25 to 6-27	●	●	●	●
BCD 4-digit multiplication, division	BCD 4-digits	B✱	B✱ S1 S2 D	(S1) × (S2) → (D + 1, D)	⌐¬	9	●			●	6-28 to 6-30	●	●	●	●
		B✱P	B✱P S1 S2 D		⌐↑	9	●			●	6-28 to 6-30	●	●	●	●
		B/	B/ S1 S2 D	(S1) ÷ (S2) → Quotient (D) Remainder (D + 1)	⌐¬	9	●			●	6-28 to 6-30	●	●	●	●
		B/P	B/P S1 S2 D		⌐↑	9	●			●	6-28 to 6-30	●	●	●	●
BCD 8-digit multiplication, division	BCD 8-digits	DB✱	DB✱ S1 S2 D	(S1 + 1, S1) × (S2 + 1, S2) → (D + 3, D + 2, D + 1, D)	⌐¬	11	●			●	6-31 to 6-33	●	●	●	●
		DB✱P	DB✱P S1 S2 D		⌐↑	11	●			●	6-31 to 6-33	●	●	●	●
		DB/	DB/ S1 S2 D	(S1 + 1, S1) ÷ (S2 + 1, S2) → Quotient (D + 1, D), Remainder (D + 3, D + 2)	⌐¬	11	●			●	6-31 to 6-33	●	●	●	●
		DB/P	DB/P S1 S2 D		⌐↑	11	●			●	6-31 to 6-33	●	●	●	●
BIN data increment	16 bits	INC	INC D	(D) + 1 → (D)	⌐¬	3	●	●		●	6-34 to 6-35	●	●	●	●
		INCP	INCP D		⌐↑	3	●	●		●	6-34 to 6-35	●	●	●	●
	32 bits	DINC	DINC D	(D + 1, D) + 1 → (D + 1, D)	⌐¬	3	●	●		●	6-36 to 6-37	●	●	●	●
		DINCP	DINCP D		⌐↑	3	●	●		●	6-36 to 6-37	●	●	●	●
BIN data decrement	16 bits	DEC	DEC D	(D) − 1 → (D)	⌐¬	3	●	●		●	6-34 to 6-35	●	●	●	●
		DECP	DECP D		⌐↑	3	●	●		●	6-34 to 6-35	●	●	●	●
	32 bits	DDEC	DDEC D	(D + 1, D) − 1 → (D + 1, D)	⌐¬	3	●	●		●	6-36 to 6-37	●	●	●	●
		DDECP	DDECP D		⌐↑	3	●	●		●	6-36 to 6-37	●	●	●	●

Table 4.15 Arithmetic Operation Instructions

(3) BCD ⟷ BIN conversion instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
BCD conversion	16 bits	BCD	BCD S D	BCD conversion (S) → (D) ↑ BIN (0 to 9999)	⌐¬	5	●	●		●	6-39 to 6-41	●	●	●	●
		BCDP	BCDP S D		⌐↑	5	●	●		●	6-39 to 6-41	●	●	●	●

Table 4.16 BCD ⟷ BIN Conversion Instructions (Continue)

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
BCD conversion	32 bits	DBCD	DBCD (S) (D)	BCD conversion (S1 + 1, S1) → (D + 1, D); BIN (0 to 99999999)		9	●			●	6-39 to 6-41	●	●	●	●
		DBCDP	DBCDP (S) (D)			9	●			●	6-39 to 6-41	●	●	●	●
BIN conversion	4-digits	BIN	BIN (S) (D)	BIN conversion (S) → (D); BCD (0 to 9999)		5	●	●		●	6-42 to 6-44	●	●	●	●
		BINP	BINP (S) (D)			5	●	●		●	6-42 to 6-44	●	●	●	●
	8-digits	DBIN	DBIN (S) (D)	BIN conversion (S1 + 1, S1) → (D + 1, D); BCD (0 to 99999999)		9	●			●	6-42 to 6-44	●	●	●	●
		DBINP	DBINP (S) (D)			9	●			●	6-42 to 6-44	●	●	●	●

Table 4.16 BCD ⟷ BIN Conversion Instructions

(4) Data transfer instructions

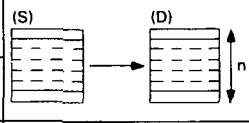
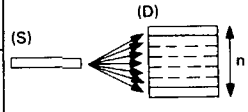
Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Transfer	16 bits	MOV	MOV (S) (D)	(S) → (D)		5	●	●		●	6-46 to 6-47	●	●	●	●
		MOVP	MOVP (S) (D)			5	●	●		●	6-46 to 6-47	●	●	●	●
	32 bits	DMOV	DMOV (S) (D)	(S + 1, S) → (D + 1, D)		7	●	●		●	6-46 to 6-47	●	●	●	●
		DMOVP	DMOVP (S) (D)			7	●	●		●	6-46 to 6-47	●	●	●	●
Negation transfer	16 bits	CML	CML (S) (D)	$\overline{(S)} \rightarrow (D)$		5	●	●		●	6-48 to 6-50	●	●	●	●
		CMLP	CMLP (S) (D)			5	●	●		●	6-48 to 6-50	●	●	●	●
	32 bits	DCML	DCML (S) (D)	$\overline{(S + 1, S)} \rightarrow (D + 1, D)$		7	●	●		●	6-48 to 6-50	●	●	●	●
		DCMLP	DCMLP (S) (D)			7	●	●		●	6-48 to 6-50	●	●	●	●
Block transfer	16 bits	BMOV	BMOV (S) (D) n			9	●			●	6-51 to 6-53	●	●	●	●
		BMOVP	BMOVP (S) (D) n			9	●			●	6-51 to 6-53	●	●	●	●
		FMOV	FMOV (S) (D) n			9	●			●	6-51 to 6-53	●	●	●	●
		FMOVP	FMOVP (S) (D) n			9	●			●	6-51 to 6-53	●	●	●	●
Exchange	16 bits	XCH	XCH (D1) (D2)	(D1) ⟷ (D2)		5	●	●		●	6-54 to 6-55	●	●	●	●
		XCHP	XCHP (D1) (D2)			5	●	●		●	6-54 to 6-55	●	●	●	●
	32 bits	DXCH	DXCH (D1) (D2)	(D1 + 1, D1) ⟷ (D2 + 1, D2)		7	●	●		●	6-54 to 6-55	●	●	●	●
		DXCHP	DXCHP (D1) (D2)			7	●	●		●	6-54 to 6-55	●	●	●	●

Table 4.17 Data Transfer Instructions

4.2.4 Application instructions

(1) Logical operation instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Logical product	16 bits	WAND	WAND (S) (D)	(D) AND (S) → (D)		5	●	●		●	7-3 to 7-6	●	●	◐	●
		WANDP	WANDP (S) (D)			5	●	●		●	7-3 to 7-6	●	●	◐	●
		WAND	WAND (S1) (S2) (D)	(S1) AND (S2) → (D)		7	●			●	7-3 to 7-6	●	●	◐	●
		WANDP	WANDP (S1) (S2) (D)			7	●			●	7-3 to 7-6	●	●	◐	●
	32 bits	DAND	DAND (S) (D)	(D + 1, D) AND (S + 1, S) → (D + 1, D)		9	●			●	7-3 to 7-6	●	●	◐	●
		DANDP	DANDP (S) (D)			9	●			●	7-3 to 7-6	●	●	◐	●
Logical sum	16 bits	WOR	WOR (S) (D)	(D) OR (S) → (D)		5	●	●		●	7-7 to 7-10	●	●	◐	●
		WORP	WORP (S) (D)			5	●	●		●	7-7 to 7-10	●	●	◐	●
		WOR	WOR (S1) (S2) (D)	(S1) OR (S2) → (D)		7	●			●	7-7 to 7-10	●	●	◐	●
		WORP	WORP (S1) (S2) (D)			7	●			●	7-7 to 7-10	●	●	◐	●
	32 bits	DOR	DOR (S) (D)	(D + 1, D) OR (S + 1, S) → (D + 1, D)		9	●			●	7-7 to 7-10	●	●	◐	●
		DORP	DORP (S) (D)			9	●			●	7-7 to 7-10	●	●	◐	●
Exclusive logical sum	16 bits	WXOR	WXOR (S) (D)	(D) XOR (S) → (D)		5	●	●		●	7-11 to 7-14	●	●	◐	●
		WXORP	WXORP (S) (D)			5	●	●		●	7-11 to 7-14	●	●	◐	●
		WXOR	WXOR (S1) (S2) (D)	(S1) XOR (S2) → (D)		7	●			●	7-11 to 7-14	●	●	◐	●
		WXORP	WXORP (S1) (S2) (D)			7	●			●	7-11 to 7-14	●	●	◐	●
	32 bits	DXOR	DXOR (S) (D)	(D + 1, D) XOR (S + 1, S) → (D + 1, D)		9	●			●	7-11 to 7-14	●	●	◐	●
		DXORP	DXORP (S) (D)			9	●			●	7-11 to 7-14	●	●	◐	●
NOT exclusive logical sum	16 bits	WXNR	WXNR (S) (D)	$\overline{(D) \text{ XOR } (S)} \rightarrow (D)$		5	●	●		●	7-15 to 7-18	●	●	◐	●
		WXNRP	WXNRP (S) (D)			5	●	●		●	7-15 to 7-18	●	●	◐	●
		WXNR	WXNR (S1) (S2) (D)	$\overline{(S1) \text{ XOR } (S2)} \rightarrow (D)$		7	●			●	7-15 to 7-18	●	●	◐	●
		WXNRP	WXNRP (S1) (S2) (D)			7	●			●	7-15 to 7-18	●	●	◐	●
	32 bits	DXNR	DXNR (S) (D)	$\overline{(D + 1, D) \text{ XOR } (S + 1, S)} \rightarrow (D + 1, D)$		9	●			●	7-15 to 7-18	●	●	◐	●
		DXNRP	DXNRP (S) (D)			9	●			●	7-15 to 7-18	●	●	◐	●
2's complement	16 bits	NEG	NEG (D)	$\overline{(D)} + 1 \rightarrow (D)$		3	●			●	7-19 to 7-20	●	●	◐	●
		NEGP	NEGP (S) (D)			3	●			●	7-19 to 7-20	●	●	◐	●

Table 4.18 Logical Operation Instructions

(2) Rotation instructions

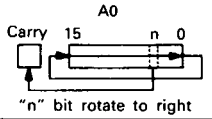
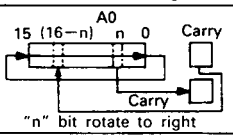
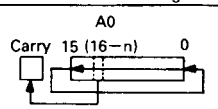
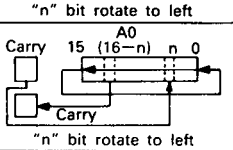
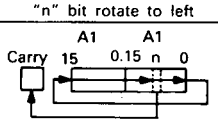
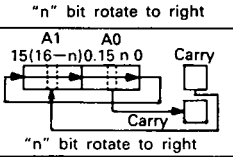
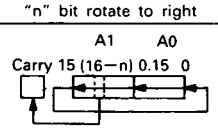
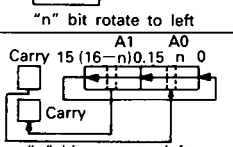
Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Right ward rotation	16 bits	ROR	ROR n			3	●		●		7-22 to 7-23	●	●	●	●
		RORP	RORP n			3	●		●		7-22 to 7-23	●	●	●	●
		RCR	RCR n			3	●		●		7-22 to 7-23	●	●	●	●
		RCRP	RCRP n			3	●		●		7-22 to 7-23	●	●	●	●
Left ward rotation		ROL	ROL n			3	●		●		7-24 to 7-25	●	●	●	●
		ROLP	ROLP n			3	●		●		7-24 to 7-25	●	●	●	●
		RCL	RCL n			3	●		●		7-24 to 7-25	●	●	●	●
		RCLP	RCLP n			3	●		●		7-24 to 7-25	●	●	●	●
Right ward rotation	32 bits	DROR	DROR n			3	●		●		7-26 to 7-27	●	●	●	●
		DRORP	DRORP n			3	●		●		7-26 to 7-27	●	●	●	●
		DRCR	DRCR n			3	●		●		7-26 to 7-27	●	●	●	●
		DRCRP	DRCRP n			3	●		●		7-26 to 7-27	●	●	●	●
Left ward rotation		DROL	DROL n			3	●		●		7-28 to 7-29	●	●	●	●
		DROLP	DROLP n			3	●		●		7-28 to 7-29	●	●	●	●
		DRCL	DRCL n			3	●		●		7-28 to 7-29	●	●	●	●
		DRCLP	DRCLP n			3	●		●		7-28 to 7-29	●	●	●	●

Table 4.19 Rotation Instructions

(3) Shift instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
n bit shift	16 bits	SFR	SFR Ⓓ n	15 n 0		5	●	●	●	●	7-31 to 7-32	●	●	●	●
		SFRP	SFRP Ⓓ n	15 to 0 Carry; 0 to 0		5	●	●	●	●	7-31 to 7-32	●	●	●	●
		SFL	SFL Ⓓ n	15 n 0		5	●	●	●	●	7-31 to 7-32	●	●	●	●
		SFLP	SFLP Ⓓ n	15 to 0 Carry; 0 to 0		5	●	●	●	●	7-31 to 7-32	●	●	●	●
1 bit shift	n bit	BSFR	BSFR Ⓓ n	n (D)		7	●		●	●	7-33 to 7-34	●	●	●	●
		BSFRP	BSFRP Ⓓ n	to 0 Carry		7	●		●	●	7-33 to 7-34	●	●	●	●
		BSFL	BSFL Ⓓ n	n (D)		7	●		●	●	7-33 to 7-34	●	●	●	●
		BSFLP	BSFLP Ⓓ n	to 0 Carry		7	●		●	●	7-33 to 7-34	●	●	●	●
1 ward shift	n ward	DSFR	DSFR Ⓓ n	n (D)		7	●			●	7-35 to 7-36	●	●	●	●
		DSFRP	DSFRP Ⓓ n	to 0		7	●			●	7-35 to 7-36	●	●	●	●
		DSFL	DSFL Ⓓ n	n (D)		7	●			●	7-35 to 7-36	●	●	●	●
		DSFLP	DSFLP Ⓓ n	to 0		7	●			●	7-35 to 7-36	●	●	●	●

Table 4.20 Shift Instructions

(4) Data processing instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
Data search	16 bits	SER	SER (S1) (S2) n	(S1) (S2) n; A0 : Coinciding number A1 : Coinciding quantity		9	●			●	7-38 to 7-39	●	●	●	●
		SERP	SERP (S1) (S2) n			9	●			●	7-38 to 7-39	●	●	●	●
Bit check		SUM	SUM (S)	(S) 15 0 A0 : Quantity of 1		3	●			●	7-40 to 7-41	●	●	●	●
		SUMP	SUMP (S)			3	●			●	7-40 to 7-41	●	●	●	●
	32 bits	DSUM	DSUM (S)	(S+1) (S) A0 : Quantity of 1		3	●			●	7-40 to 7-41	●	●	●	●
		DSUMP	DSUMP (S)			3	●			●	7-40 to 7-41	●	●	●	●
Decode Encode	2^n bits	DECO	DECO (S) (D) n	8 → 256 decode (S) (D) n Decode 2^n bits		9	●			●	7-42 to 7-43	●	●	●	●
		DECOP	DECOP (S) (D) n			9	●			●	7-42 to 7-43	●	●	●	●
		ENCO	ENCO (S) (D) n	256 → 8 encode (S) 2^n bits Encode (D) n		9	●			●	7-42 to 7-43	●	●	●	●
		ENCOP	ENCOP (S) (D) n			9	●			●	7-42 to 7-43	●	●	●	●
7 segment decode	16 bits	SEG	SEG (S) (D)	(S) 3 0 7SEG (D) n 0 Valid for A□N, A3H when M9052 is off. Decode		7	●				7-44 to 7-45	●	●	●	●
Bit set reset		BSET	BSET (D) n	(D) 15 n 0 1		7	●			●	7-46 to 7-47	●	●	●	●
		BSETP	BSETP (D) n			7	●			●	7-46 to 7-47	●	●	●	●
		BRST	BRST (D) n	(D) 15 n 0 0		7	●			●	7-46 to 7-47	●	●	●	●
		BRSTP	BRSTP (D) n			7	●			●	7-46 to 7-47	●	●	●	●
Accociation Dissociation		DIS	DIS (S) (D) n	4 bits All 0 4 bits S D D+1 D+2 When n = 3		9	●			●	7-48 to 7-50	●	●	●	●
		DISP	DISP (S) (D) n			9	●			●	7-48 to 7-50	●	●	●	●
		UNI	UNI (S) (D) n	4 bits 4 bits S S+1 S+2 D When n = 3		9	●			●	7-48 to 7-50	●	●	●	●
		UNIP	UNIP (S) (D) n			9	●			●	7-48 to 7-50	●	●	●	●
ASCII conversion	—	ASC	ASC Alphanumeric character (D)	Converts alphanumeric characters into ASCII codes and stores into 4 points beginning with the device, D.		13	●			●	7-51 to 7-52	●	●	●	●

Table 4.21 Data Processing Instructions

(5) FIFO instructions

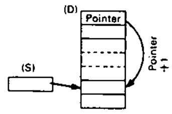
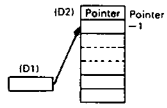
Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Write	16 bits	FIFW	FIFW (S) (D)			7	●			●	7-54 to 7-57	●	●	●	●
		FIFWP	FIFWP (S) (D)			7	●			●	7-54 to 7-57	●	●	●	●
Read		FIFR	FIFR (D1) (D2)			7	●			●	7-54 to 7-57	●	●	●	●
		FIFRP	FIFRP (D1) (D2)			7	●			●	7-54 to 7-57	●	●	●	●

Table 4.22 FIFO Instructions

(6) Buffer memory Access instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Data read	1 word	FROM	FROM n1 n2 (D) n3	Reads data from the special function module.		9	●			●	7-59 to 7-60	●	●	●	●
		FROMP	FROMP n1 n2 (D) n3			9	●			●	7-59 to 7-60	●	●	●	●
	2 words	DFRO	DFRO n1 n2 (D) n3			9	●			●	7-59 to 7-60	●	●	●	●
		DFROP	DFROP n1 n2 (D) n3			9	●			●	7-59 to 7-60	●	●	●	●
Data write	1 word	TO	TO n1 n2 (S) n3	Writes data to the special function module.		9	●			●	7-61 to 7-62	●	●	●	●
		TOP	TOP n1 n2 (S) n3			9	●			●	7-61 to 7-62	●	●	●	●
	2 words	DTO	DTO n1 n2 (S) n3			11	●			●	7-61 to 7-62	●	●	●	●
		DTOP	DTOP n1 n2 (D) n3			11	●			●	7-61 to 7-62	●	●	●	●

Table 4.23 Buffer Memory Access Instructions

4

(7) Local, remote I/O station access instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU A1N	A2N	A3N	A3H
Local station data read, write	1 word	LRDP	LRDP n1 (S) (D) n2	Reads data from the local station.		11	●			●	7-64 to 7-67	●	●	●	●
		LWTP	LWTP n1 (D) (S) n2	Writes data to the local station.		11	●			●	7-64 to 7-67	●	●	●	●
Remote I/O station data read, write		RFRP	RFRP n1 n2 (D) n3	Reads data from the special function module in the remote I/O station.		11	●			●	7-68 to 7-70	●	●	●	●
		RTOP	RTOP n1 n2 (D) n3	Writes data to the special function module in the remote I/O station.		11	●			●	7-68 to 7-70	●	●	●	●

Table 4.24 Local, Remote I/O Station Access Instructions

(8) Display instructions

Classification	Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
												A1N	A2N	A3N	A3H
ASCII print	—	PR	PR Ⓢ Ⓓ	Outputs ASCII codes (16 characters) from the specified devices (8 points) to the output module.		7	●			●	7-73 to 7-78	—	—	—	—
		PR	PR Ⓢ Ⓓ	Outputs ASCII codes sequentially from the specified devices to the output module until NUL (00H) is given.		7	●			●	7-73 to 7-78	●	●	●	●
		PRC	PRC Ⓢ Ⓓ	Converts the comment in the specified device into ASCII code and outputs to the output module. The comment in device 1 may be output.		7	●				7-73 to 7-78	●	●	●	●
Display	—	LED	LED Ⓢ	(S) to (S)+7 → Display 16 characters		3	●			●	7-79 to 7-81	—	—	●	●
		LEDA	LEDA Alphanumeric character	Indicates the specified alphanumeric characters on the display. (LEDA: First 8 characters LEDB: Second 8 characters)		13					7-82 to 7-83	—	—	●	●
		LEDB	LEDB Alphanumeric character			13					7-82 to 7-83	—	—	●	●
		LEDC	LEDC Ⓢ	Displays the comment in device, S.		3	●			●	7-79 to 7-81	—	—	●	●
Display reset	—	LEDR	LEDR	Reset the display indication.		1					7-84 to 7-85	●	●	●	●

Table 4.25 Display Instructions

(9) Other instructions

Classification		Unit	Instruction Symbol	Symbol	Contents of Processing	Execution Condition	Number of steps	Index	Subset	Carry flag	Error flag	Page	Applicable CPU			
													A1N	A2N	A3N	A3H
WDT reset		—	WDT	WDT	WDT is reset in sequence program		1					7-87 to 7-88	●	●	●	●
		—	WDTP	WDTP			1					7-87 to 7-88	●	●	●	●
Status latch	set	—	SLT	SLT	At the condition set by parameter setting, data are stored into memory for status latch.		1					7-89 to 7-90	—	●	●	●
	reset		SLTR	SLTR	Status latch is reset and SLT instruction is enabled		1					7-89 to 7-90	—	●	●	●
Sampling trace	set	—	STRA	STRA	At the condition set by parameter setting, sampling data are stored into memory for status latch.		1					7-91 to 7-92	—	●	●	●
	reset		STRAR	STRAR	Sampling trace is resumed. (STRA instruction is enabled.)		1					7-91 to 7-92	—	●	●	●
Carry	set	1 bit	STC	STC	Carry flag contact (M9012) is turned on.		1					7-93 to 7-94	●	●	●	●
	reset		CLC	CLC	Carry flag contact (M9012) is turned off.		1					7-93 to 7-94	●	●	●	●
Timing clock		1 bit	DUTY	DUTY n1 n2 Ⓓ	Timing clock shown below is generated. (D) n1 scan n2 scan		7				●	7-95 to 7-96	●	●	●	●

Table 4.26 Other Instructions

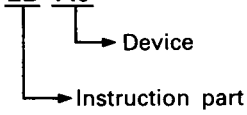
4.3 Instruction Structure

1) Many instructions may be divided into an instruction part and a device as follows:
   { Instruction part·····Indicates the function.
   { Device·····Indicates the data for use with that instruction.

2) The instruction structure may be largely classified as follows with the instruction part and device(s) combined:
   a) [Instruction part] ····· Retains the device status and mainly controls the program.
      Example: END, FEND

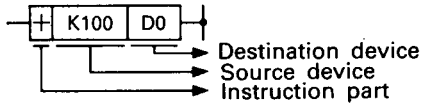
   b) [Instruction part] + [Device] ····· Switches the device on/off, controls the execution condition in accordance with the device status, branches the program, etc.

      Example: LD X0

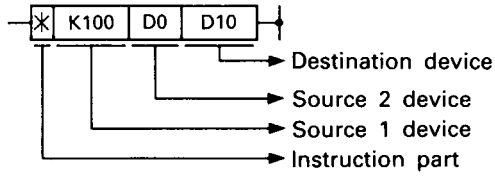
   c) [Instruction part] + [Source device] + [Destination device] ····· Operation is performed using the destination data and source data, and the operation result is stored to the destination.

      Example:

   d) [Instruction part] + [Source 1 device] + [Source 2 device] + [Destination device] ····· Operation is performed using the source 1 data and source 2 data, and the operation result is stored to the destination.

      Example:

   e) Others ····· Combination of a) to d).

(1) Source (S)

1) Source data is used for operation.
2) Source data depends on the device specified as follows:
   • Constant ····· Specify the numeric value used for the operation. This value is set while the program is being written and cannot be changed during run of the program.

   • Bit device, word device ·····Specify the device which stores the data used for the operation. Hence, the data must be stored to the specified device before the operation is initiated. By changing the data to be stored to the specified device during program run, the data used with the instruction can be changed.

(2) Destination (D)

1) Stores data after operation is performed. When the instruction consists of [instruction part] + [source device] + [destination device], the data used for the operation must be stored to the destination before the operation is started.
2) The device for storing data must be specified at the destination.

REMARKS

1) In this manual, the sources and destination are represented as follows:

Source ····· (S)
Source 1 ····· (S1)
Source 2 ····· (S2)
Destination ····· (D)

4.4 Bit Processing

Bit processing is performed when a bit device (X, Y, M, L, S, B, F) has been specified. 1-bit processing or digit-specification processing may be selected.

4.4.1 1-bit processing

When the sequence instruction is used, more than one bit (one point) cannot be specified for the bit device.

Example: LD, X0, OUT Y20

4.4.2 Digit specification processing

When the basic and application instructions are used, the number of digits may need to be specified for the bit device. Up to 16 points can be specified in 4 point increments when a 16-bit instruction is used, and up to 32 points can be specified when a 32-bit instruction is used.

(a) 16-bit instruction: K1 to 4 (4 to 16 points)

Example: Setting range by the digit specification of 16-bit data, X0 to F

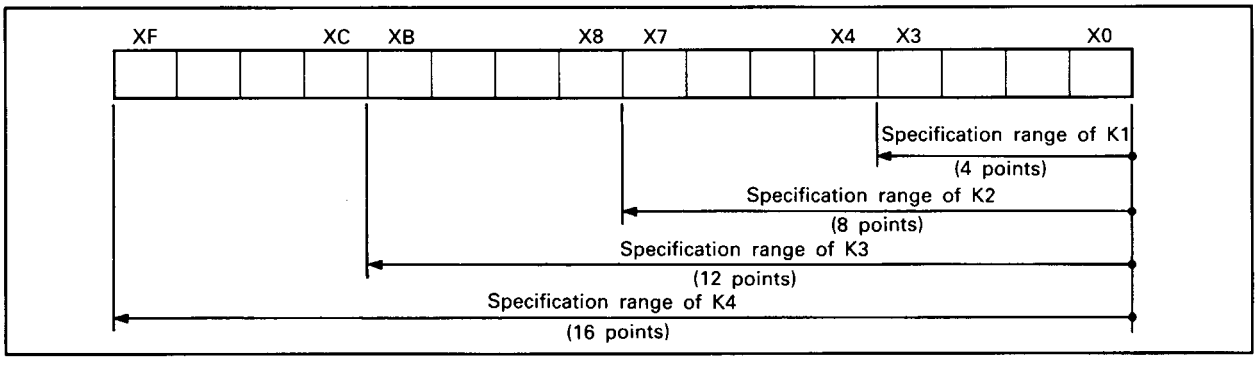


Fig. 4.4 Digit Specification Range of 16-Bit Instruction

(b) 32-bit instruction: K1 to 8 (4 to 32 points)

Example: Setting range by the digit specification of 32-bit data, X0 to 1F

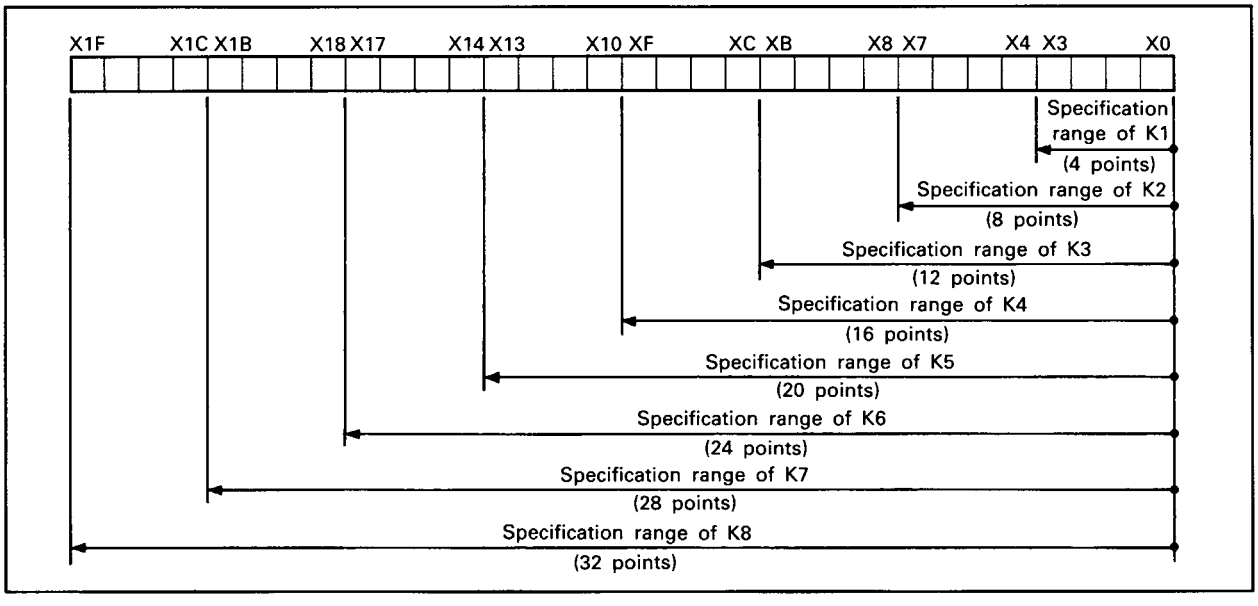


Fig. 4.5 Digit Specification Range of 32-Bit Instruction

4.5 Word Processing

Word processing indicates the processing of a device word by word. 16 bits (1 word) or 32 bits (2 words) may be selected.

4.5.1 16-bit processing

1 point may be specified for the source and destination when the word device (e.g. D, W, R) is used. Max. 16 points may be specified in 4 point increments using digit specification when the bit device (e.g. X, Y, M, B, F) is used.

(1) When there is digit specification on the source (S) side, the range of numeric values handled as source data are as shown in Table 4.27.

Specified Number of Digits	16-Bit Instruction
K1 (4 points)	0 to 15
K2 (8 points)	0 to 255
K3 (12 points)	0 to 4095
K4 (16 points)	−32768 to 32767

Table 4.27 List of Digit Specification and Handled Numeric Values

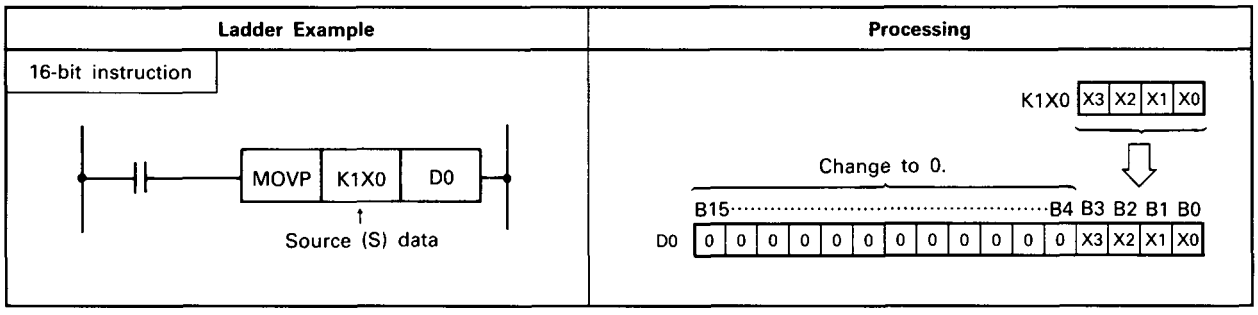


Fig. 4.6 Ladder Example and Processing

(2) When there is digit specification on the destination (D) side, the number of points set by the digit specification is used on the destination side.

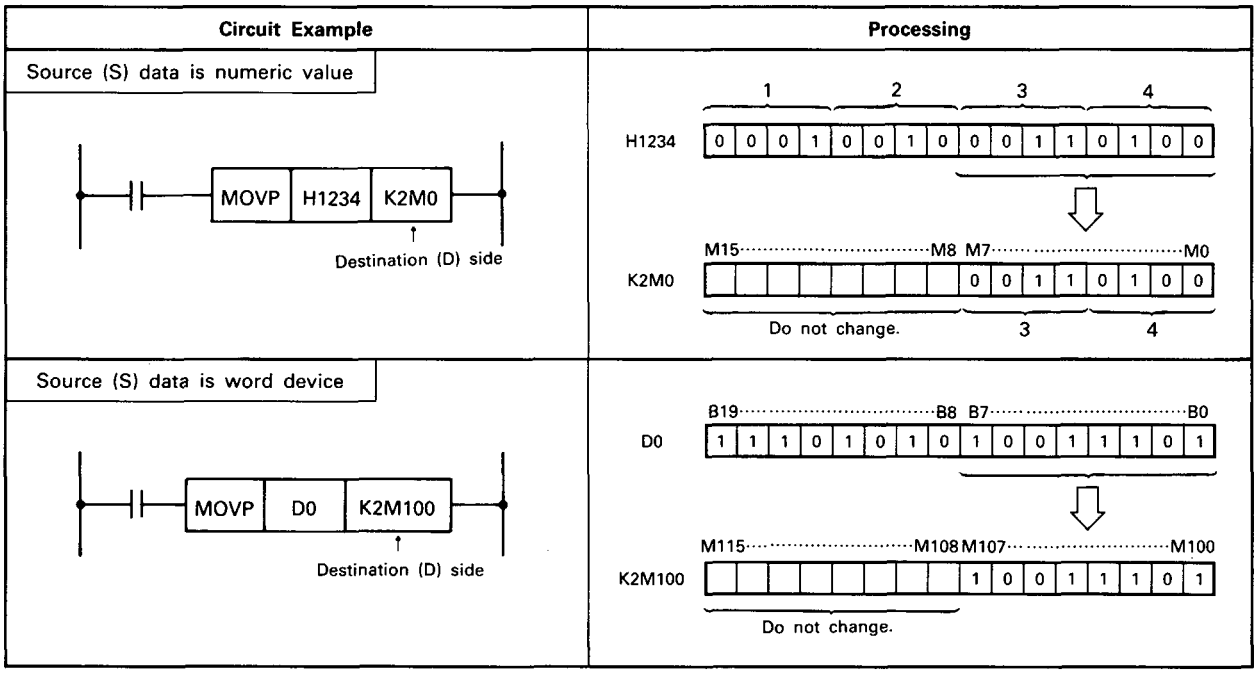


Fig. 4.7 Ladder Example and Processing

4.5.2 32-bit processing

2 points may be specified for the source and destination when the word device (e.g. D, W, R) is used. Max. 32 points may be specified in 4 point increments using digit specification when the bit device (e.g. X, Y, M, B, F) is used.

(1) When there is digit specification on the source (S) side, the range of numeric values handled as source data are as shown in Table 4.28.

Specified Number of Digits	32-Bit Instruction	Specified Number of Digits	32-Bit Instruction
K1 (4 points)	0 to 15	K5 (20 points)	0 to 1048575
K2 (8 points)	0 to 255	K6 (24 points)	0 to 167772165
K3 (12 points)	0 to 4095	K7 (28 points)	0 to 268435455
K4 (16 points)	0 to 65535	K8 (32 points)	−2147483648 to 2147483648

Table 4.28 List of Digit Specification and Handled Numeric Values

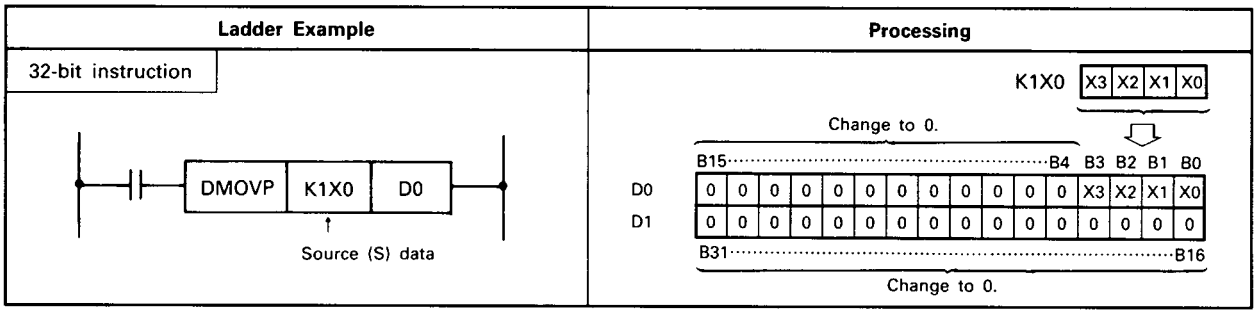


Fig. 4.8 Ladder Example and Processing

(2) When there is digit specification on the destination (D) side, the number of points set by the digit specification is used on the destination side.

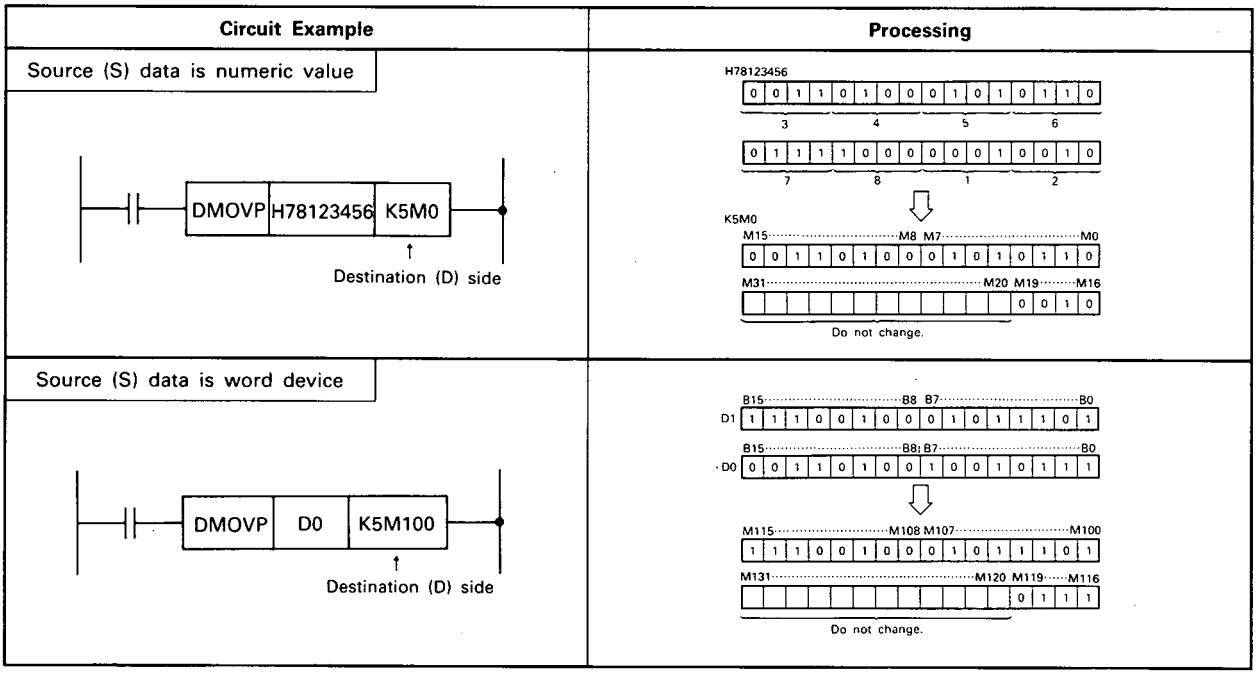


Fig. 4.9 Ladder Example and Processing

4.6 Handling of Numeric Values

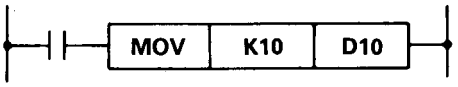
In the A series, there are instructions which handle numeric values in 16 bits and 32 bits.
The highest bits of 16 bits and 32 bits are used for the judgement of positive and negative. Therefore, numeric values handed by 16 bits and 32 bits are as follows:

16 bits: −32768 to 32767
32 bits: −2147483648 to 2147483647

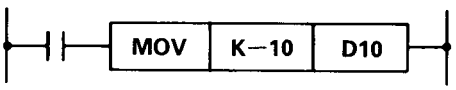
POINT

(1) Numeric value setting procedure

1) Decimal

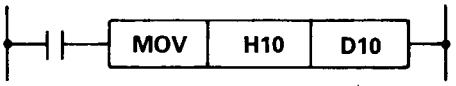


10 is stored D10 in BIN value.



−10 is stored to D10 in BIN value.

2) Hexadecimal



10 is stored to D10 in hexadecimal.

(2) When FFFEH is divided by 2, the following occurs.

16-bit instruction

Since FFFE is −2, −2 ÷ 2 = −1 (FFFFH) is stored to D0.

32-bit instruction

Since FFFE is 65534, 65534 ÷ 2 = 32767 (7FFF) is stored to D0.

When the range of numeric values handled in 16 bits and 32 bits exceeds that specified (overflow, underflow) this is indicated as in the following table.

	Processing of 16-bit Data		Processing of 32-bit Data	
	Decimal display	Hexadecimal display	Decimal display	Hexadecimal display
Overflow	32764	7 F F C H	2147483644	7 F F F F F F C H
	32765	7 F F D H	2147483645	7 F F F F F F D H
	32766	7 F F E H	2147483646	7 F F F F F F E H
	32767	7 F F F H	2147483647	7 F F F F F F F H
	Over flow −32768	8 0 0 0 H	Over flow −2147483648	8 0 0 0 0 0 0 0 H
	−32767	8 0 0 1 H	−2147483647	8 0 0 0 0 0 0 1 H
	−32766	8 0 0 2 H	−2147483646	8 0 0 0 0 0 0 2 H
	−32765	8 0 0 3 H	−2147483645	8 0 0 0 0 0 0 3 H
	⋮		⋮	
	Processing of 16-bit Data		Processing of 32-bit Data	
	Decimal display	Hexadecimal display	Decimal display	Hexadecimal display
Underflow	−32765	8 0 0 3 H	−2147483645	8 0 0 0 0 0 0 3 H
	−32766	8 0 0 2 H	−2147483646	8 0 0 0 0 0 0 2 H
	−32767	8 0 0 1 H	−2147483647	8 0 0 0 0 0 0 1 H
	−32768	8 0 0 0 H	−2147483648	8 0 0 0 0 0 0 0 H
	Under flow 32767	7 F F F H	Under flow 2147483647	7 F F F F F F F H
	32766	7 F F E H	2147483646	7 F F F F F F E H
	32765	7 F F D H	2147483645	7 F F F F F F D H
	32764	7 F F C H	2147483644	7 F F F F F F C H
	⋮		⋮	

Table 4.29 Processing Outside the Allowed Numeric Value Range

REMARKS

Even in the case of overflow and underflow, the carry flag and error flag do not change.

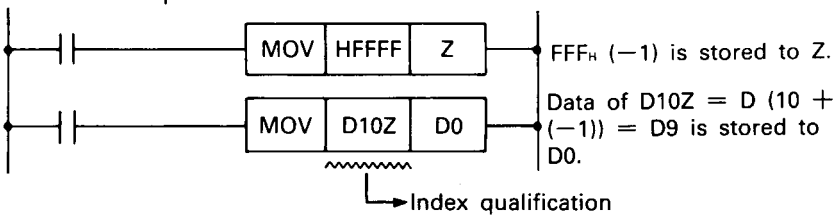
4.7 Index Qualification

(1) The index qualification is used to specify the device number be providing an index (Z, V) to the device and adding the specified device number and index content.

(2) The index qualification can be used for devices X, Y, M, L, B, F, T, C, D, R, W, K, and H.

(3) The indexes (Z, V) are provided with a sign and can be set in the range of −32768 and 32767.

(4) The index qualification is as shown below.



Example
When the index qualification is performed, the actual processing devices are as shown below.
(Z = 20, V = −5)

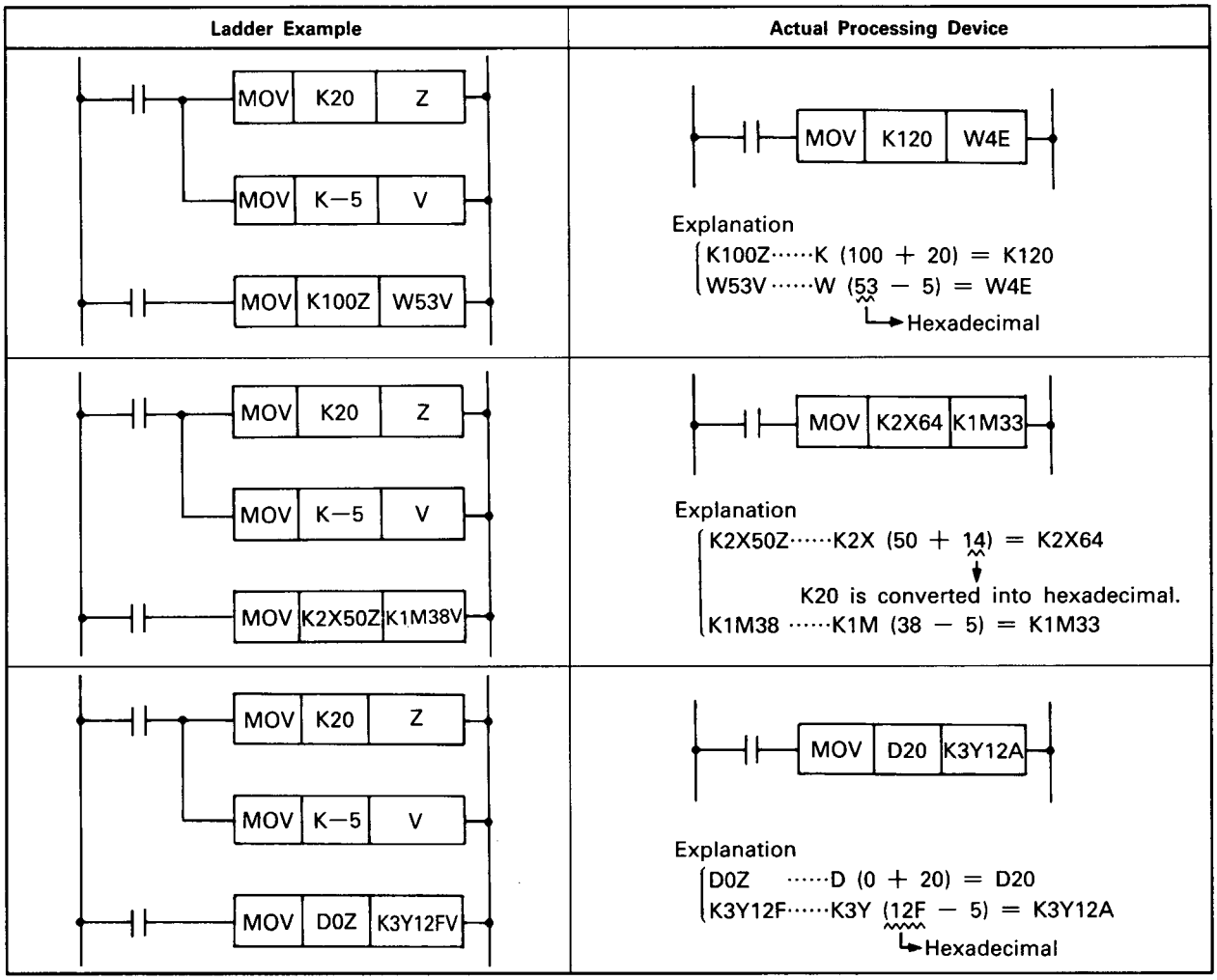


Fig. 4.10 Ladder Examples and Actual Devices Processed

4.8 Subset Processing

Increases processing speed within the following limits when the bit device is used with the basic and application instructions:

1) The bit device specified is a multiple of 8 (a multiple of 16 when the A3HCPU is used).
2) Index qualification is not performed.
3) 4K or K8 is specified when using the digit specification.

4.9 Operation Error

(1) In the following cases, the basic instruction and application instruction result in operation error.

1) Error described in the explanation of each instruction has occurred.
2) When the index qualification is performed and the device range has been exceeded. In this case, however, K and H are excluded.

Index	Circuit Example	Judgement
Z = −10	MOV K−10 Z MOV T9Z D0	Since T(9 + (−10)) = T − 1, operation error occurs.
Z = 10	MOV K10 Z MOV K1020Z K4Y30	Since D (1020 + 10) = D1030 and the range of D0 to 1024 is exceeded, operation error occurs.
Z = 10	MOV K10 Z MOV K32367Z D0	Since K (32767 + 10) = K − 32759, operation error does not occur.

Fig. 4.11 Ladder Examples and Judgements

3) When the index qualification is performed and the head number of bit device has exceeded the corresponding device range.

Index	Circuit Example	Judgement
Z = 15	MOV K15 Z MOV K4B3F0Z D0 Link relay	Although K4B3FF (B(3F0 + F) = B3FF) is specified, operation error does not occur.
Z = 16	MOV K16 Z MOV K4B3F0Z D0	Since K4B400 (B(3F0 + 10) = B400) is specified and the corresponding device range is exceeded, operation error occurs.

Fig. 4.12 Ladder Examples and Judgements

POINT

If the specified range of a device has exceeded the allowable device range, data will be written to the next device. Therefore, caution should be exercised.

MOV D0 K4B3F8 — Although B3F8 to 407 have been specified, B400 to 407 to not exist.

DMOV D0 W3FF — Although W3FF and 400 have been specified, W400 does not exist actually.

(2) Error processing

1) If an operation error has occurred during the execution of basic instructions or application instructions, the error flag (M9010, 9011) is turned on and the error step number is stored into the error step storage register (D9010, 9011).

Error flag
- M9010·····Turned on by operation error and turned off when the next basic instruction or application instruction is valid.
- M9011·····Turned on and latched by the first operation error.

Error step storage register
- D9010·····Stores the head step number of the instruction which has caused the operation error.
- D9011·····Stores the head step number of instruction which has caused operation error first. The stored step number is latched.

2) D9011 stores the step number of the instruction which has caused an operation error when M9011 changes from off to on. Therefore, if M9011 remains on, the contents of D9011 do not change.

3) Program the reset of M9011 and D9011 as shown below.

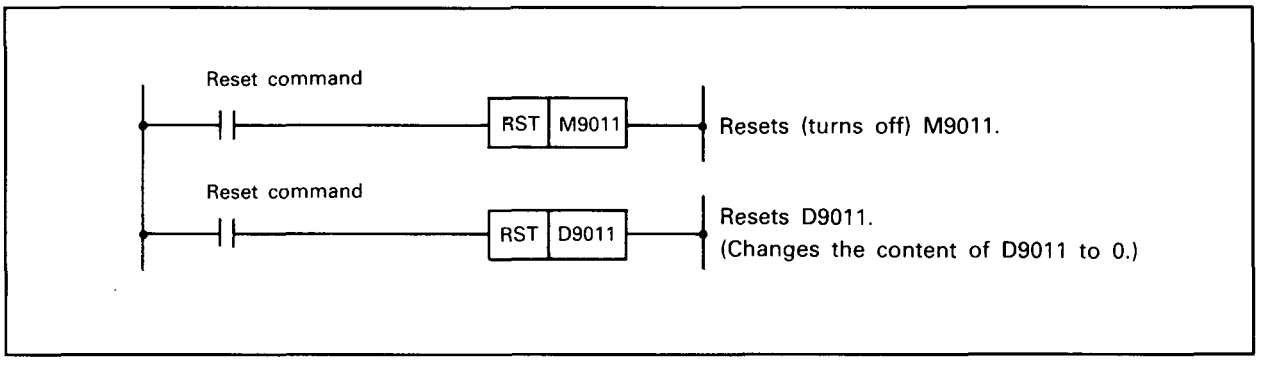
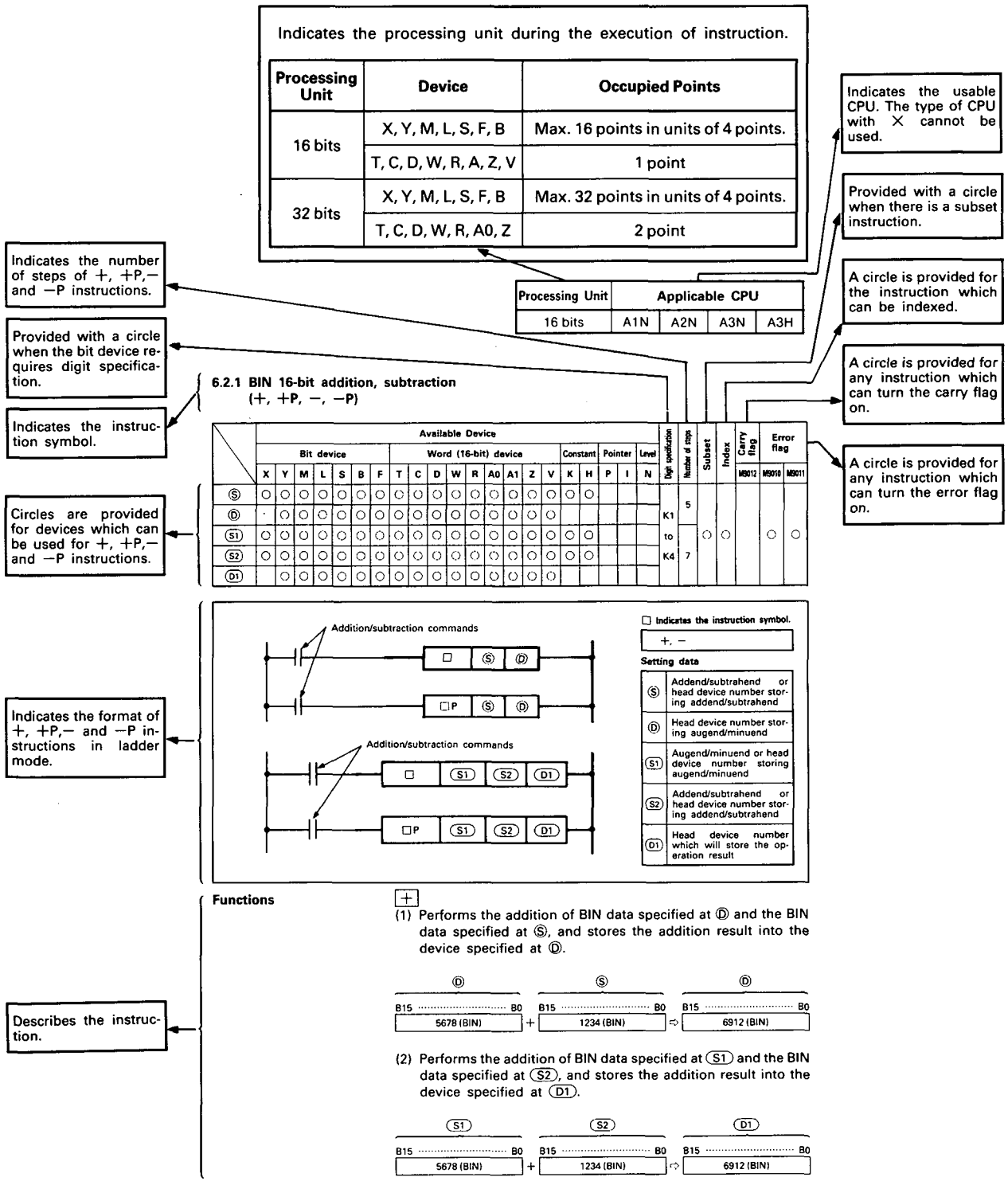


Fig. 4.13 Resetting the Special Relay, Register

4) If an operation error has occurred, sequence processing may be stopped or continued as selected by the parameter setting (Section 3.5.16).

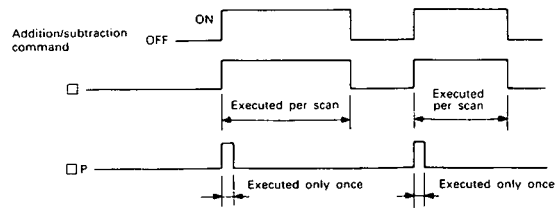
4.10 Instruction Format

The following sections give explanations of sequence instructions, basic instructions, and application instructions. They are given in the following format.



Indicates the execution conditions of +, +P, − and −P instructions.

Execution Conditions

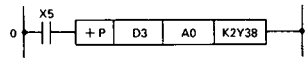


Indicates the program examples of +, +P, − and −P instructions.

Program Examples

(+)

Program which adds the content of A0 to the content of D3 and outputs the result to Y38 to 3F when X5 turns on.

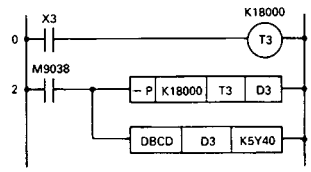


Coding

Step Number	Instruction	Device			
0	LD	X5			
1	+P	D3	A0	K2Y38	
8	END				

(−)

Program which outputs the difference between the set value and present value of timer T3 to Y40 to 4F in BCD.



Coding

Step Number	Instruction	Device			
0	LD	X3			
1	OUT	T3	K18000		
2	LD	M9036			
3	−P	K18000	T3	D3	
10	DBCD	D3	K5Y40		
19	END				

4

5. SEQUENCE INSTRUCTIONS

Sequence instructions are classified as follows:

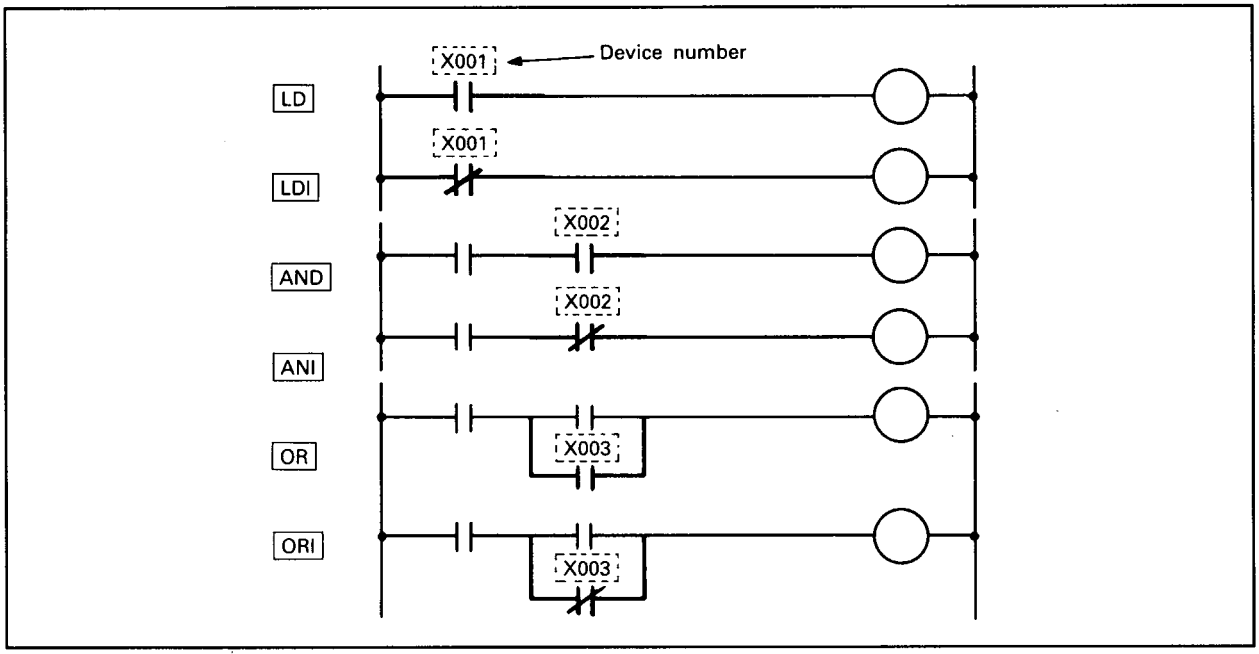
Classification	Description	Refer to:
Contact instruction	Operation start, series connection, parallel connection	5-2 to 5-4
Connection instruction	Ladder block series connection, parallel connection, operation result storage	5-5 to 5-13
Output instruction	Bit device output, differential output, set, reset, output reverse	5-14 to 5-25
Shift instruction	Bit device shift	5-26 to 5-27
Master control instruction	Master control set, reset	5-28 to 5-29
Program branch instruction	Jump, call, interrupt enable, disable	5-30 to 5-40
Program switching instruction	Switching between main and subprograms	5-41 to 5-48
FOR ~ NEXT instruction	FOR ~ NEXT	5-49 to 5-50
Refresh instruction	Data link refresh, I/O partial refresh	5-51 to 5-56
Termination instruction	Sequence program termination	5-57 to 5-60
Other instruction	Sequence program stop, no operation	5-61 to 5-64

5.1 Contact Instructions

5.1.1 Operation start, series connection, parallel connection (LD, LDI, AND, ANI, OR, ORI)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
○	○	○	○	○	○	○	○	○														1					



Functions

LD, LDI

(1) LD is an "N/O" contact operation start instruction and LDI is a "NK" contact operation start instruction. (These instructions draw the ON/OFF data of the specified device and use the data as an operation result.)

AND, ANI

(1) AND is an "N/O" contact serial connection instruction and ANI is a "N/C" contact serial connection instruction. (These instructions draw the ON/OFF data of the specified device, performs the AND operation of that data and the previous operation result, and use it as a new operation result.

(2) There are no restrictions on the use of AND and ANI. In ladder mode on the A6GPP, A6PHP, A6HGP, however, there are the following restrictions:

1) Write: When AND or ANI is connected serially, a circuit of up to 21 stages can be created.

2) Read: When AND or ANI is connected serially, a circuit of up to 24 stages can be displayed at one time.

OR, ORI

(1) OR is a one "N/O" contact parallel connection instruction and ORI is a one "N/C" contact parallel connection instruction. (These instructions draw the ON/OFF data of specified device, performs the OR operation of that data and the previous operation result, and use it as an operation result.)

(2) There are no restrictions on the use of OR and ORI. In ladder mode on the A6GPP, A6PHP, A6HGP, however, there are the following restrictions:

1) Write: A circuit, in which up to 23 ORs or ORIs are connected consecutively, may be written.

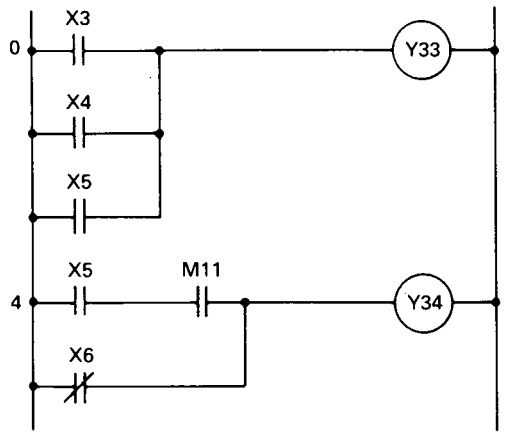
2) Read: A circuit, in which up to 23 ORs or ORIs are connected consecutively, may be displayed. A circuit containing more than 23 ORs or ORIs cannot be completely displayed.

Execution Conditions Executed every scan independently of the device status and operation result.

IB (NA) 66147-A

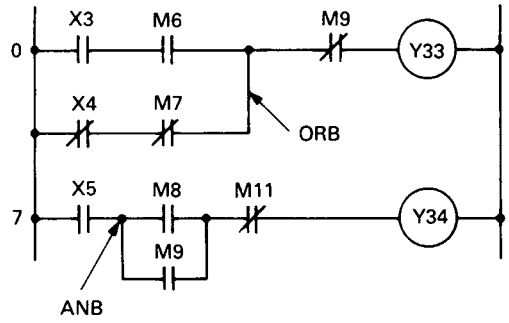
Program Examples

LD, LD2, AND, ANI, OR, ORI



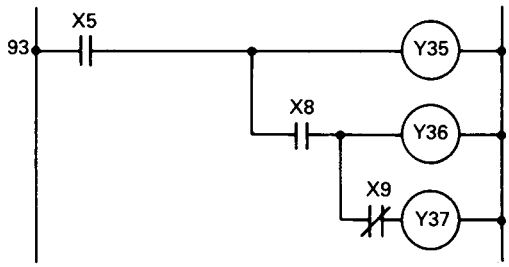
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	OR	X4			
2	OR	X5			
3	OUT	Y33			
4	LD	X5			
5	AND	M11			
6	ORI	X6			
7	OUT	Y34			
8	END				



Coding

Step Number	Instruction	Device			
0	LD	X3			
1	AND	M6			
2	LDI	X4			
3	ANI	M7			
4	ORB				
5	ANI	M9			
6	OUT	Y33			
7	LD	X5			
8	LD	M8			
9	OR	M9			
10	ANB				
11	ANI	M11			
12	OUT	Y34			
13	END				



Coding

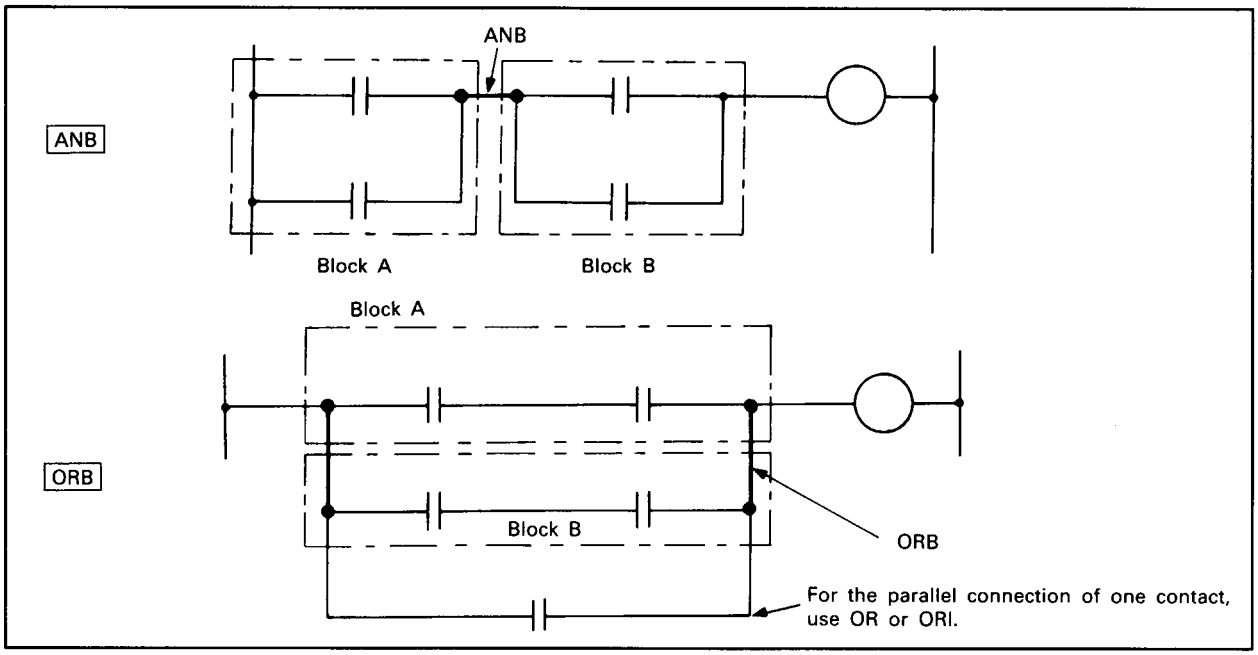
Step Number	Instruction	Device			
93	LD	X5			
94	OUT	Y35			
95	AND	X8			
96	OUT	Y36			
97	ANI	X9			
98	OUT	Y37			
99	END				

5.2 Connection Instructions

5.2.1 Ladder block series connection, parallel connection (ANB, ORB)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

ANB

(1) This instruction performs the AND operation of block A and block B, and uses it as an operation result.

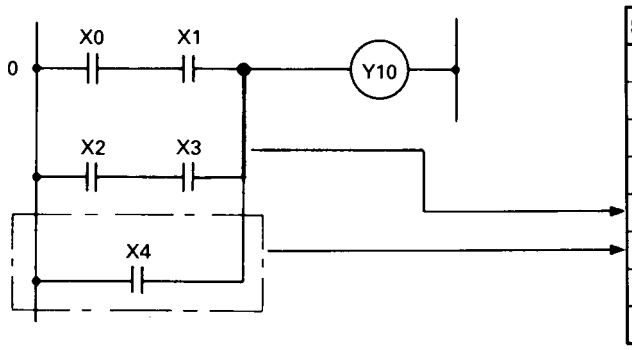
(2) The symbol of ANB is not a contact symbol but a connection symbol.

(3) When ANB is written consecutively, a maximum of seven instructions (eight blocks) can be written. If eight or more instructions are written consecutively, the PC cannot perform proper operation. (Refer to Coding Example 2 on the next page.)

ORB

(1) This instruction performs the OR operation of block A and block B, and uses it as an operation result.

(2) ORB performs parallel connection of circuit blocks with two or more contacts. For parallel connection of circuit blocks which have only one contact, OR and ORI are used and ORB is not required. (See below.)



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	AND	X1			
2	LD	X2			
3	AND	X3			
4	ORB				
5	OR	X4			
6	OUT	Y10			
7	END				

(3) The symbol of ORB is not a contact symbol but a connection symbol.

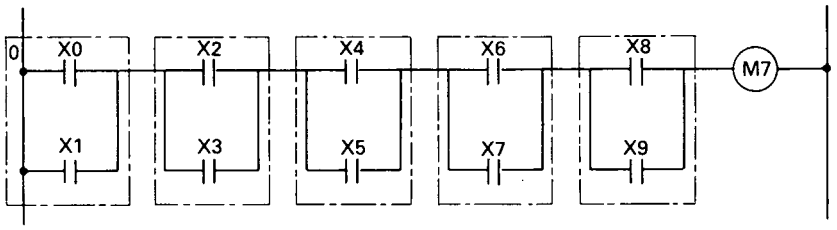
(4) When ORB is written consecutively, a maximum of seven instructions (eight blocks) can be written. If eight or more instructions are written consecutively, the PC cannot perform proper operation. (Refer to Coding Example 2 on the next page.)

IB (NA) 66147-A

Program Examples

ANB

When circuit blocks are serially connected consecutively, the coding of program is available in two types. However, proceed with the coding according to Coding example 1.



Coding example 1

Step Number	Instruction	Device			
0	LD	X0			
1	OR	X1			
2	LD	X2			
3	OR	X3			
4	ANB				
5	LD	X4			
6	OR	X5			
7	ANB				
8	LD	X6			
9	OR	X7			
10	ANB				
11	LD	X8			
12	OR	X9			
13	ANB				
14	OUT	M7			
15	END				

There is no restriction on the number of ANBs used.

Coding example 2

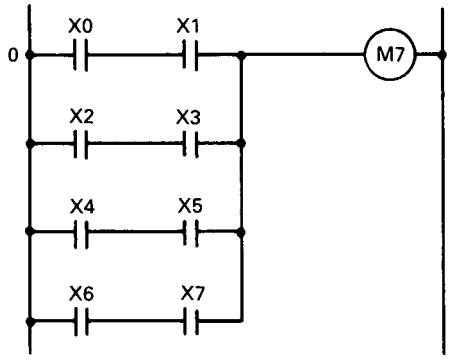
Step Number	Instruction	Device			
0	LD	X0			
1	OR	X1			
2	LD	X2			
3	OR	X3			
4	LD	X4			
5	OR	X5			
6	LD	X6			
7	OR	X7			
8	LD	X8			
9	OR	X9			
10	ANB				
11	ANB				
12	ANB				
13	ANB				
14	OUT	M7			
15	END				

Continuous write of ANB is allowable to a maximum of seven instructions (eight blocks). If eight or more instructions are written consecutively, the PC cannot perform proper operation.

5

ORB

When circuit blocks are parallelly connected consecutively, the coding of program is available in two types. However, proceed with the coding according to Coding example 1.



Coding example 1

Step Number	Instruction	Device			
0	LD	X0			
1	AND	X1			
2	LD	X2			
3	AND	X3			
4	ORB				
5	LD	X4			
6	AND	X5			
7	ORB				
8	LD	X6			
9	AND	X7			
10	ORB				
11	OUT	M7			
12	END				

There is no restriction on the number of ORBs used.

Coding example 2

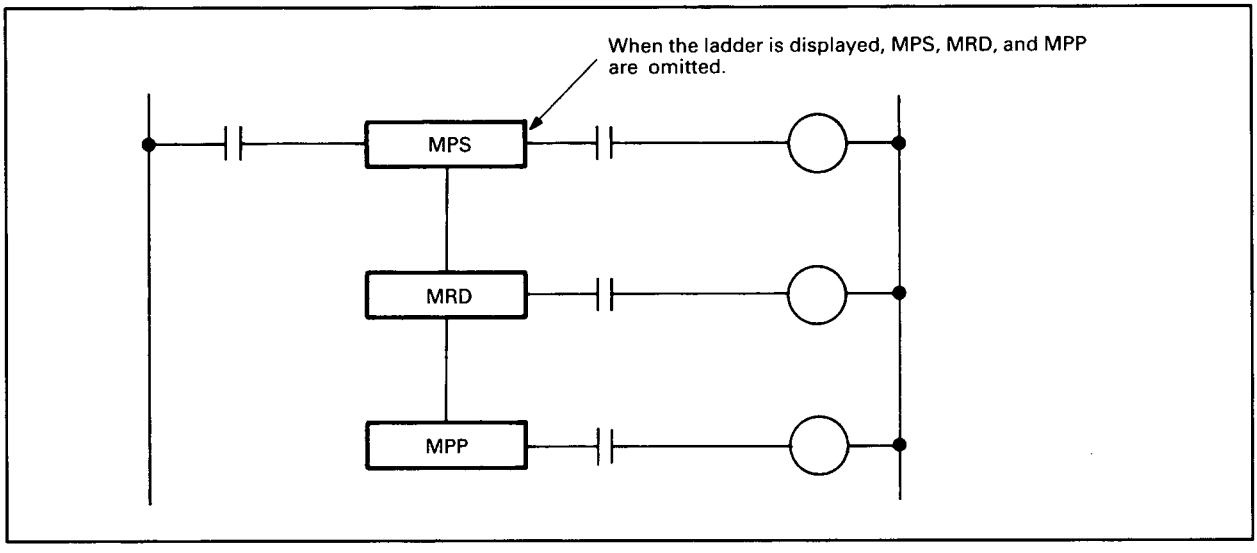
Step Number	Instruction	Device			
0	LD	X0			
1	AND	X1			
2	LD	X2			
3	AND	X3			
4	LD	X4			
5	AND	X5			
6	LD	X6			
7	AND	X7			
8	ORB				
9	ORB				
10	ORB				
11	OUT	M7			
12	END				

Continuous write of ORB is allowable to a maximum of seven instructions (eight blocks). If eight or more instructions are written consecutively, the PC cannot perform proper operation.

5.2.2 Operation result push, read, pop (MPS, MRD, MPP)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

MPS

(1) Stores the operation result (ON/OFF) immediately preceding the MPS instruction.

(2) The MPS instruction can be used a maximum of 11 times consecutively. However, if an MPP instruction is used in between, 1 is reduced from the number of used MPS instructions.

MRD

(1) Reads the operation result stored by the MPS instruction, and resumes the operation with that operation result, starting at the next step.

5

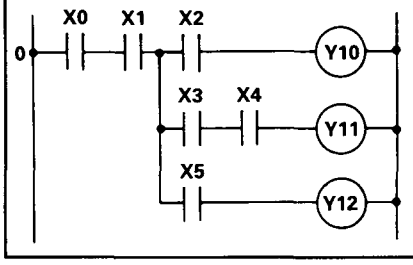
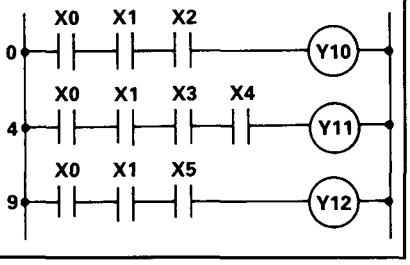
MPP

(1) Reads the operation result stored by the MPS instruction, and resumes the operation with that operation result, starting at the next step.

(2) Clears the operation result stored by the MPS instruction.

POINT

(1) When MPS, MRD, and MPP are used and when they are not used, the circuits differ as shown below.

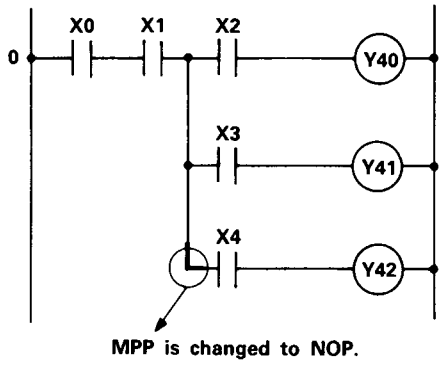
Circuit Using MPS, MRD, and MPP	Circuit Not Using MPS, MRD, and MPP
	

POINT

(2) Set the numbers of used MPS and MPP instructions to the same. If the used numbers differ, the following occurs.

1) When the number of MPS instructions is larger than that of MPP instructions, the PC performs operation in the changed circuit.

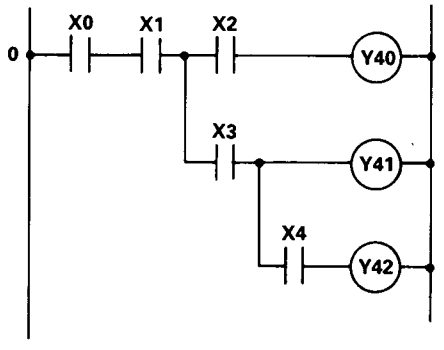
Before change



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	AND	X1			
2	MPS				
3	AND	X2			
4	OUT	Y40			
5	MRD				
6	AND	X3			
7	OUT	Y41			
8	MPP				
9	AND	X4			
10	OUT	Y42			
11	END				

After change



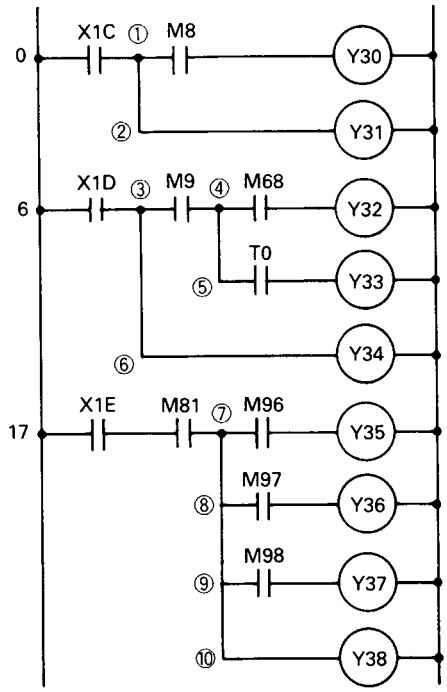
Coding

Step Number	Instruction	Device			
0	LD	X0			
1	AND	X1			
2	MPS				
3	AND	X2			
4	OUT	Y40			
5	MRD				
6	AND	X3			
7	OUT	Y41			
8	MPP				
9	AND	X4			
10	OUT	Y42			
11	END				

2) If the number of MPP instructions is larger than that of MPS instructions, this results in circuit plotting error and the PC cannot perform proper operation.

Program Examples

1) Program which uses MPS, MRD, and MPP.



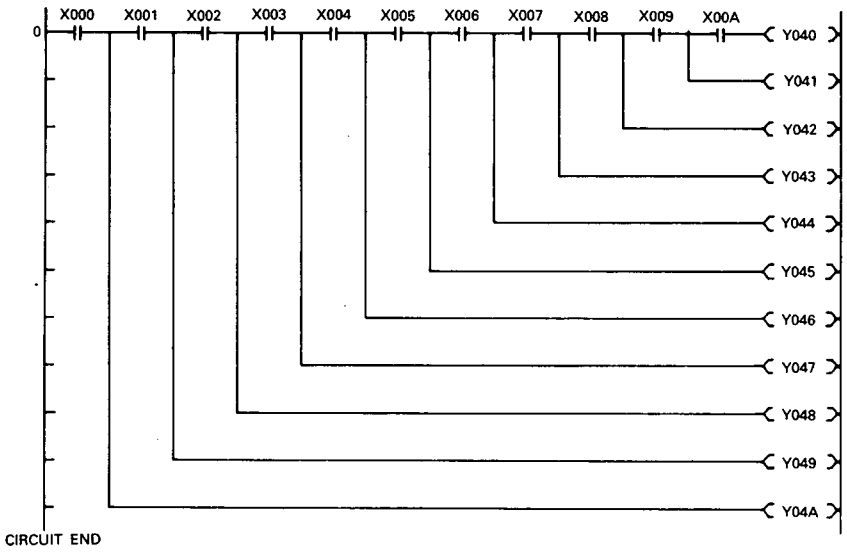
Coding

	Step Number	Instruction	Device			
①	0	LD	X1C			
	1	MPS				
	2	AND	M8			
	3	OUT	Y30			
②	4	MPP				
	5	OUT	Y31			
	6	LD	X1D			
③	7	MPS				
	8	ANI	M9			
④	9	MPS				
	10	AND	M68			
	11	OUT	Y32			
⑤	12	MPP				
	13	AND	T0			
	14	OUT	Y33			
⑥	15	MPP				
	16	OUT	Y34			
	17	LD	X1E			
	18	AND	M81			
⑦	19	MPS				
	20	AND	M96			
	21	OUT	Y35			
⑧	22	MRD				
	23	AND	M97			
	24	OUT	Y36			
⑨	25	MRD				
	26	AND	M98			
	27	OUT	Y37			
⑩	28	MPP				
	29	OUT	Y38			
	30	END				

5

2) Printing example by use of MPS and MPP instructions.

○ Circuit printing



○ List printing

0	LD	X000	32	MPP	
1	MPS		33	OUT	Y046
2	AND	X001	34	MPP	
3	MPS		35	OUT	Y047
4	AND	X002	36	MPP	
5	MPS		37	OUT	Y048
6	AND	X003	38	MPP	
7	MPS		39	OUT	Y049
8	AND	X004	40	MPP	
9	MPS		41	OUT	Y04A
10	AND	X005	42	END	
11	MPS		43	NOP	
12	AND	X006	44	NOP	
13	MPS		45	NOP	
14	AND	X007	46	NOP	
15	MPS		47	NOP	
16	AND	X008	48	NOP	
17	MPS		49	NOP	
18	AND	X009	50	NOP	
19	MPS		51	NOP	
20	AND	X00A	52	NOP	
21	OUT	Y040	53	NOP	
22	MPP		54	NOP	
23	OUT	Y041	55	NOP	
24	MPP		56	NOP	
25	OUT	Y042	57	NOP	
26	MPP		58	NOP	
27	OUT	Y043	59	NOP	
28	MPP		60	NOP	
29	OUT	Y044	61	NOP	
30	MPP		62	NOP	
31	OUT	Y045	63	NOP	

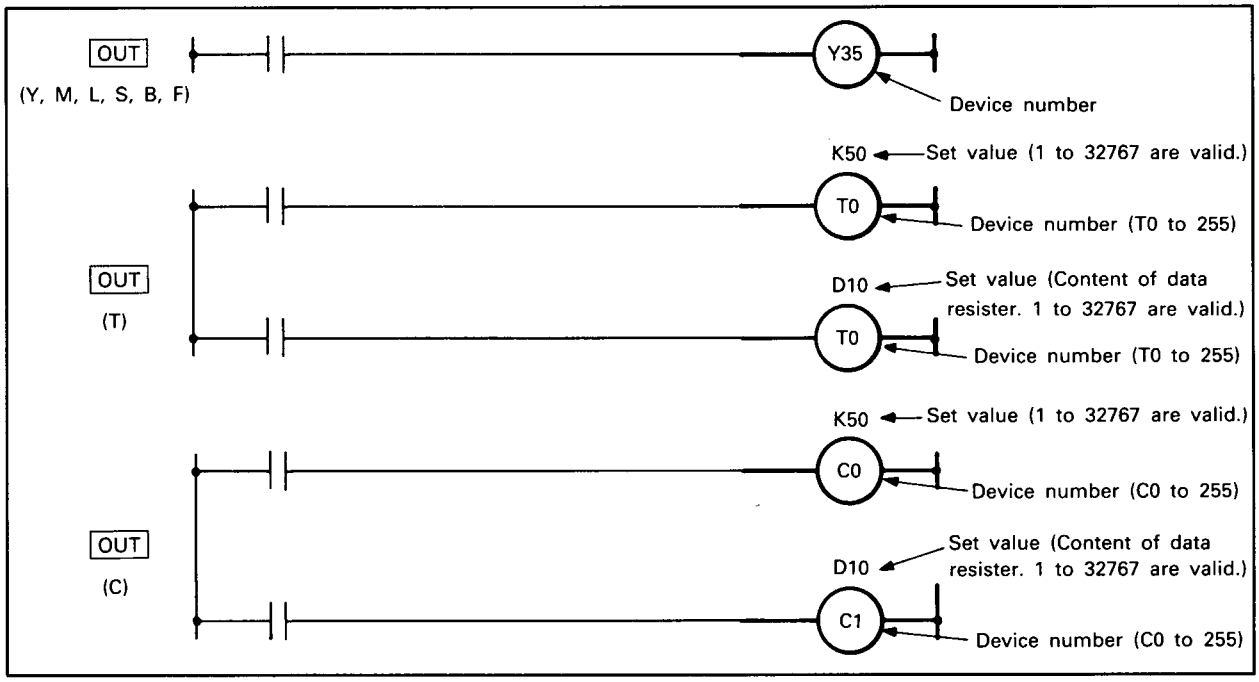
MEMO

5.3 Output Instructions

5.3.1 Bit device, timer, counter output (OUT)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Bit device			○	○	○	○	○	○																1/3					
Timer	Device								○															1					
	Set value										○							○											
Counter	Device									○														1					
	Set value										○							○											



Functions

OUT (Y, M, L, S, B, F)

(1) This instruction outputs the operation result for the elements preceding the OUT instruction.

Operation Result	OUT Instruction		
	Coil	Contact	
		"N/O" contact	"N/C" contact
OFF	OFF	Non-continuity	Continuity
ON	ON	Continuity	Non-continuity

(2) When OUT F□ turns on, following occurs:

1) A3N, A3HCPU: The turned-on F number if displayed at the LED indicator and the F number is stored into D9009 and 9125.

2) A1N, A2N: The "ERROR" LED flickers and the F number is stored into D9009 and 9125.

OUT (T)

(1) When the operation result of instructions preceding the OUT instruction are on, the coil of timer turns on and counts up to the set value. When the timer times up (counted value = set value), the contact is as indicated below.

"N/O" contact	Continuity
"N/C" contact	Non-continuity

(2) When the operation result of instructions preceding the OUT instruction change from ON to OFF, the following occurs.

Type of Timer	Timer Coil	Present Value of Timer	Before Time-Up		After Timer-Up	
			"N/O"	"N/C"	"N/O"	"N/C"
100ms timer	OFF	0	Non-continuity	Continuity	Non-continuity	Continuity
10ms timer						
100ms retentive	OFF	Present value is retained	Non-continuity	Continuity	Continuity	Non-continuity

(3) After the timer has timed up, the status of the contact of an integrating timer does not change until the RST instruction is executed.

(4) A negative number (−32768 to −1) cannot be set as a set value. When the set value is 0, the same processing as for 1 is performed.

(5) For the mode of operation of the timer, refer to Section 2.6.1.

OUT (C)

(1) When the operation result of the instructions preceding the OUT instruction have changed from OFF to ON, 1 is added to the present value (count value). When the counter has counted up (counted value = set value), the stale of the contact is as indicated below.

"N/O" contact	Continuity
"N/C" contact	Non-continuity

(2) When the operation result of the instructions preceding the OUT instruction remain on, counting is not performed. (It is not necessary to convert the count input into a pulse.)

(3) After the counter has counted up, the count value and the status of contact do not change until the RST instruction is executed.

(4) A negative number (−32768 to −1) cannot be used as a set value. When the set value is 0, the same processing as for 1 is performed.

(5) For the operating mode of the counter, refer to Section 2.2.6.

Execution Conditions

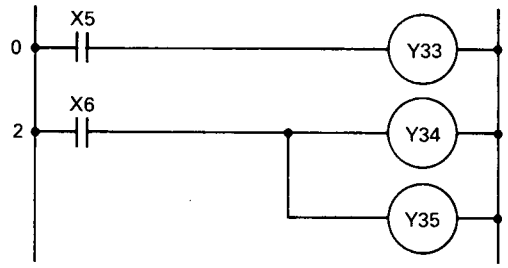
This instruction is executed per scan irrespective of the operation result of the instructions preceding the OUT instruction.

REMARKS

Three steps are employed only when the special relay is used for the device of OUT instruction.

Program Examples

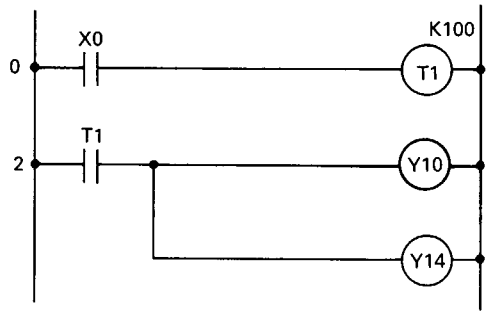
1) Program which switches an output at the output unit.



Coding

Step Number	Instruction	Device			
0	LD	X5			
1	OUT	Y33			
2	LD	X6			
3	OUT	Y34			
4	OUT	Y35			
5	END				

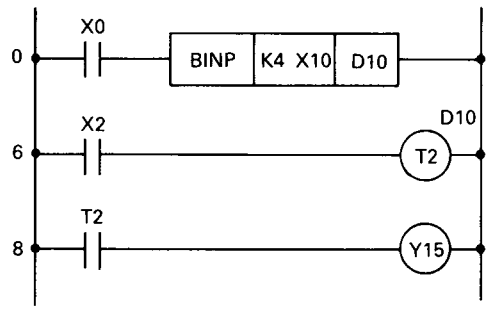
2) Program which turns on Y10 and Y14 10 seconds after X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	T1	K100		
2	LD	T1			
3	OUT	Y10			
4	OUT	Y14			
5	END				

3) Program which uses the BCD data of X10 to 1F as the set value of the timer.



Data of X10 to 1F is converted into BIN and stored into D10.

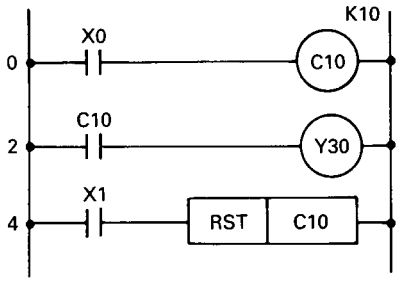
When X2 turns on, the data stored in D10 is counted as a set value.

When T2 counts up, Y15 turns on.

Coding

Step Number	Instruction	Device			
0	LD	X0			
1	BINP	K4X10	D10		
6	LD	X2			
7	OUT	T2	D10		
8	LD	T2			
9	OUT	Y15			
10	END				

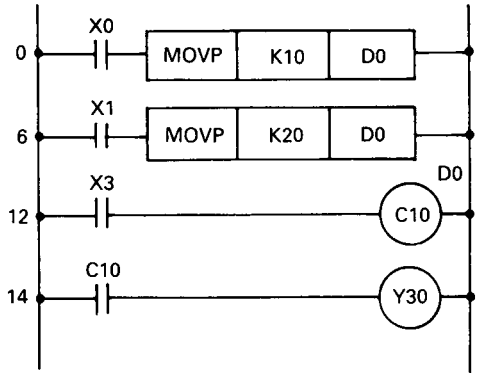
4) Program which turns on Y30 after X0 turns on 10 times and which turns off Y30 when X1 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	C10	K10		
2	LD	C10			
3	OUT	Y30			
4	LD	X1			
5	RST	C10			
8	END				

5) Program which changes the set value of C10 to 10 when X0 turns on and to 20 when X1 turns on.



When X0 turns on, 10 is stored to D0.

When X1 turns on, 20 is stored to D0.

C10 counts the data, which is stored in D0, as a set value.

When C10 counts up, Y30 turns on.

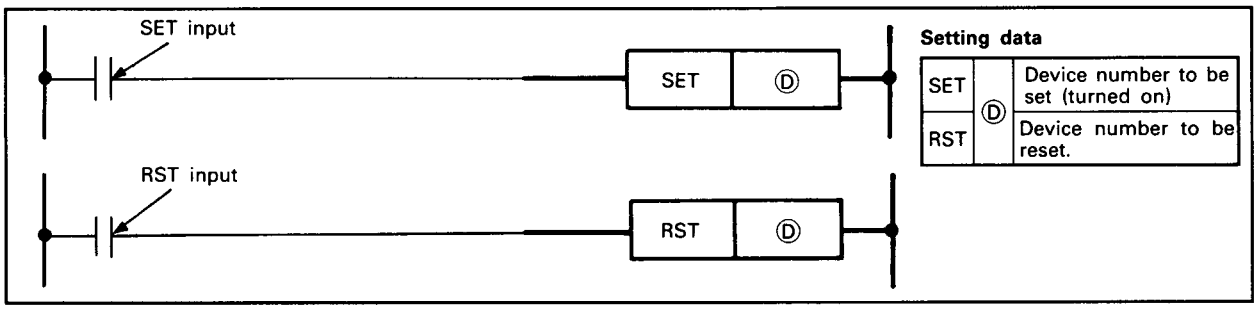
Coding

Step Number	Instruction	Device			
0	LD	X0			
1	MOVP	K10	D0		
6	LD	X1			
7	MOVP	K20	D0		
12	LD	X3			
13	OUT	C10	D0		
14	LD	C10			
15	OUT	Y30			
16	END				

5.3.2 Bit device set, reset (SET, RST)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag		
		Bit device							Word (16-bit) device									Constant		Pointer		Level								
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
SET	Ⓓ		○	○	○	○	○	○																1					
RST	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○							3					



Setting data

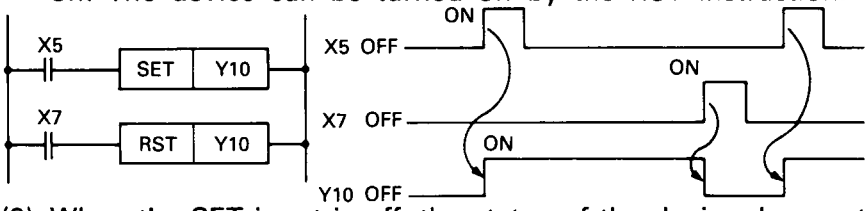
SET	Ⓓ	Device number to be set (turned on)
RST		Device number to be reset.

Functions

SET

(1) When the SET input turns on, the specified device is turned on.

(2) The turned-on device remains on even if the SET input turns off. The device can be turned off by the RST instruction.



(3) When the SET input is off, the status of the device does not change.

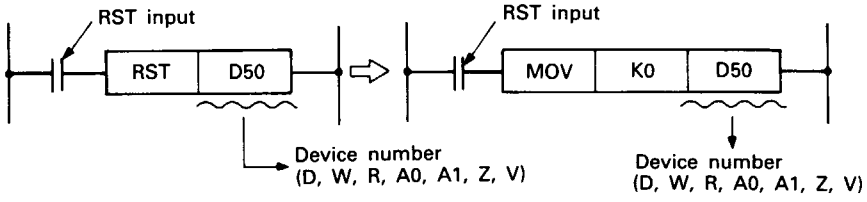
RST

(1) When the RST input turns on, the specified device changes as described below:

Device	Status
Y, M, L, S, B, F	Coil and contact are turned off.
T, C	Present value is set to 0, and coil and contact are turned off.
D, W, R, A0, A1, Z, V	Content is set to 0.

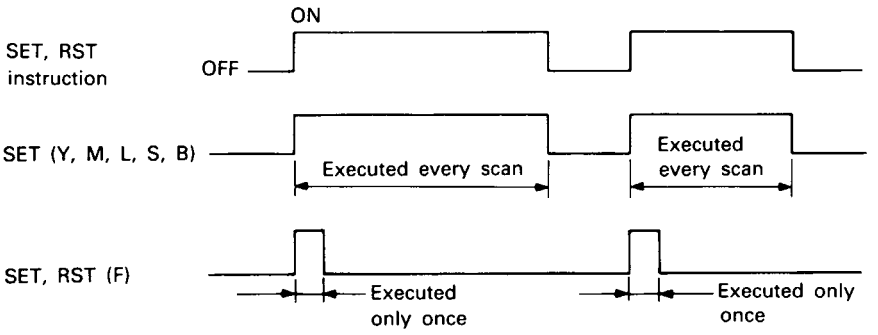
(2) When the RST input is off, the status of device does not change.

(3) The functions of RST (D, W, R, A0, A1, Z, V) are the same as those of the following circuit.

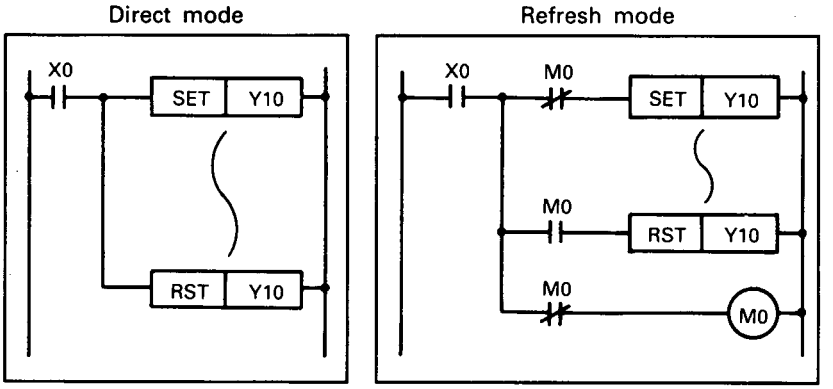


Execution Conditions

(1) The SET, RST instructions are executed on the following conditions:



(2) SET, RST instructions
In refresh mode, the SET/RST instructions cannot be used in a program which outputs a pulse signal during one scan.
In this case, output (Y) must be changed to direct mode or the program must be corrected so that the device is switched on/off every scan as shown below.



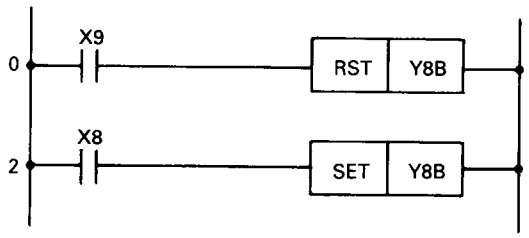
REMARKS

The number of steps is 3 when any of the following devices is used:

SET instruction	Special relay	: M9000 to M9255
	Link relay	: B0 to B3FF
	Annunciator	: F0 to F255
RST instruction	Special relay	: M9000 to F255
	Word devices	: All

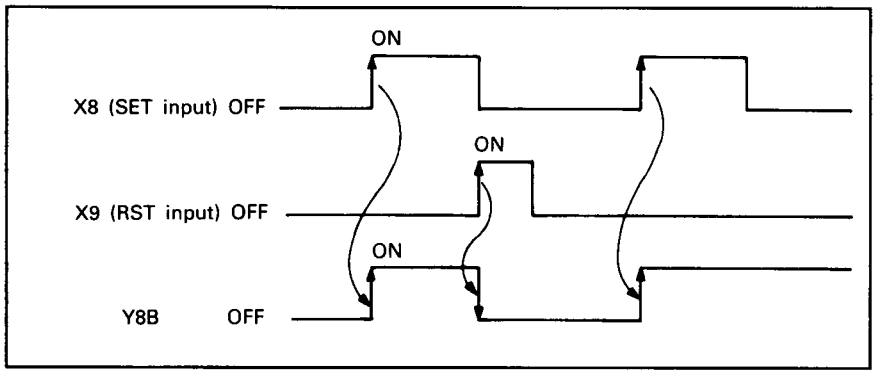
Program Examples

1) Program which sets (turns on) Y8B when X8 turns on and which resets (turns off) Y8B when X9 turns on.

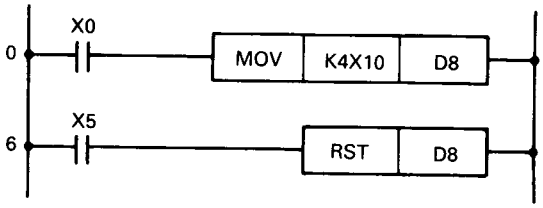


Coding

Step Number	Instruction	Device			
0	LD	X9			
1	RST	Y8B			
2	LD	X8			
3	SET	Y8B			
4	END				



2) Program which sets the content of data register to 0.



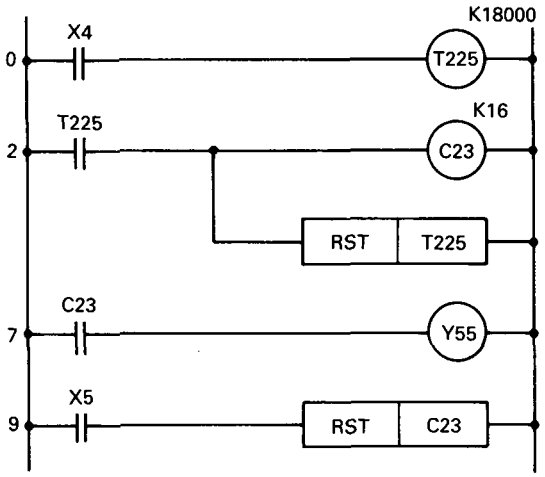
When X0 turns on, the contents of X10 to 1F are stored into D8.

When X5 turns on, the content of D8 is set to 0.

Coding

Step Number	Instruction	Device			
0	LD	X0			
1	MOV	K4X10	D8		
6	LD	X5			
7	RST	D8			
10	END				

3) Program which resets the 100ms retentive timer and counter.



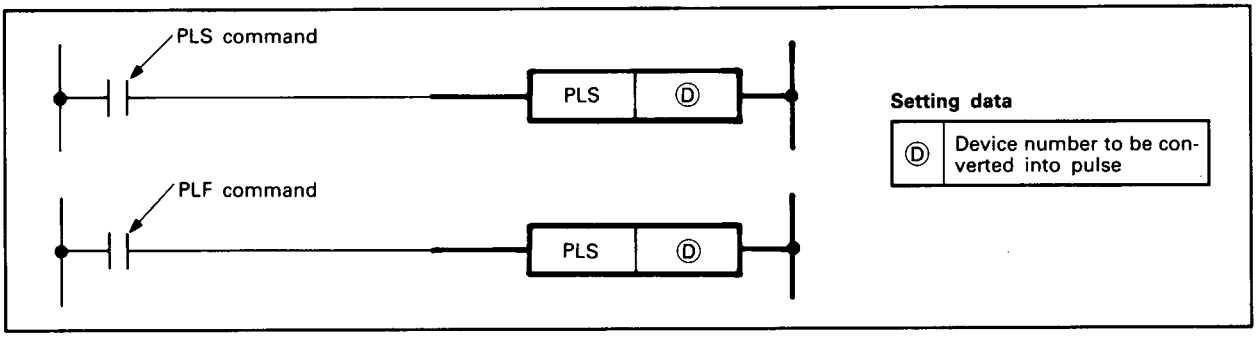
Coding

Step Number	Instruction	Device			
0	LD	X4			
1	OUT	T225	K18000		
2	LD	T225			
3	OUT	C23	K16		
4	RST	T225			
7	LD	C23			
8	OUT	Y55			
9	LD	X5			
10	RST	C23			
13	END				

5.3.3 Edge-triggered differential output (PLS, PLF)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

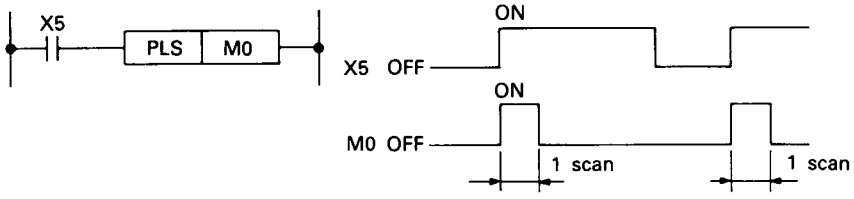
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○																3					



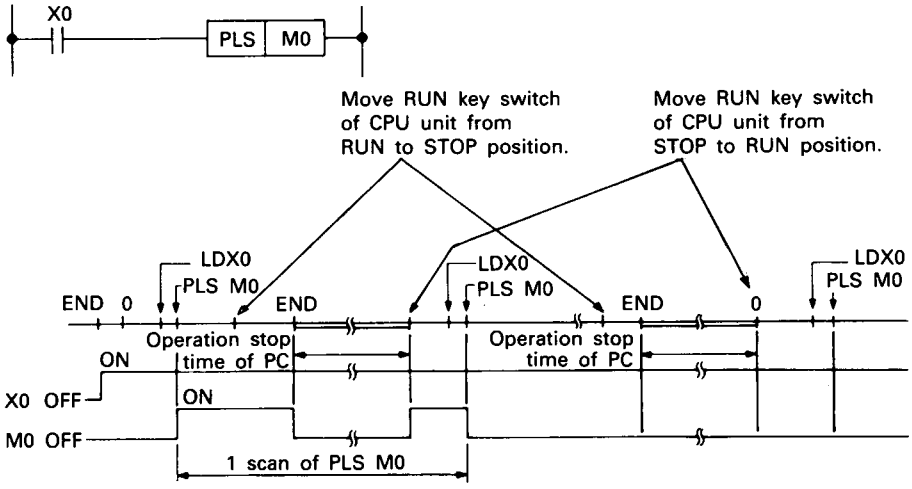
Functions

PLS

(1) The PLS instruction turns on the specified device for one scan when the PLS instruction turns from off to on. Similarly, when a device is in the on state, the PLS instruction will turn it off for one scan time.



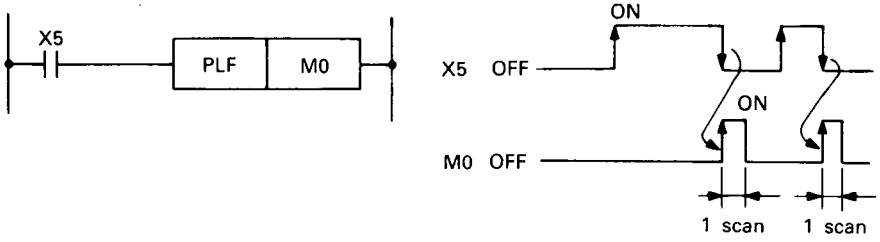
(2) If the instruction generating the pulse is switched on and the RUN key switch is moved from the RUN to STOP position and the RUN key switch is moved from the RUN to STOP position and then returned to the RUN position again, the PLS instruction is not executed.



(3) When the latch relay (L) is used with the PLS instruction, the previous data is re-output after the power is restored.

PLF

(1) The PLF instruction turns on the specified device during one scan when the PLF instruction turns from on to off. Similarly, when the device is in the on state, the PLF instruction will turn it off for one scan time.

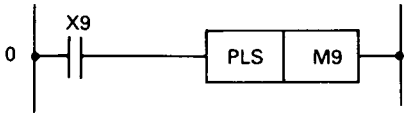


(2) If the instruction generating the pulse is off and the RUN key switch is moved from the RUN to STOP position and then returned to the RUN position again, the PLF instruction is not executed.

Program Examples

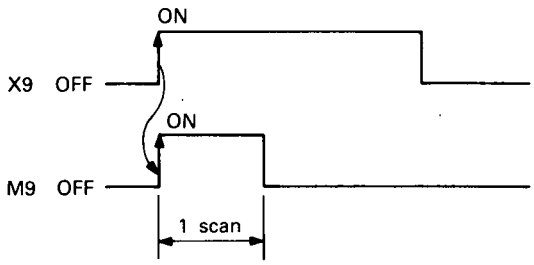
PLS

Program which executes the PLS instruction when M0 turns on.



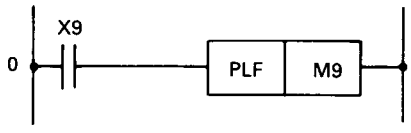
Coding

Step Number	Instruction	Device			
0	LD	X9			
1	PLS	M9			
4	END				



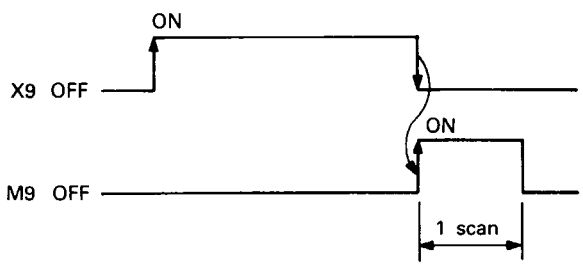
PLF

Program which executes the PLF instruction when M0 turns off.



Coding

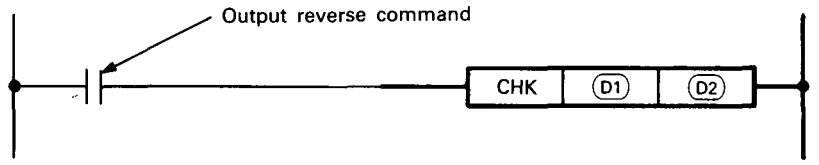
Step Number	Instruction	Device			
0	LD	X9			
1	PLF	M9			
4	END				



5.3.4 Bit device output reverse (CHK) (Valid in refresh mode)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	~~A3H~~

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(D1)		○	○	○	○	○	○																5					
(D2)		△	△	△	△	△	△	△	△	△	△	△	△	△	△	△												



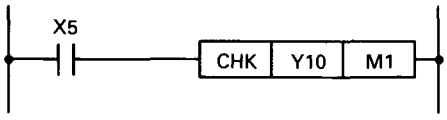
Setting data

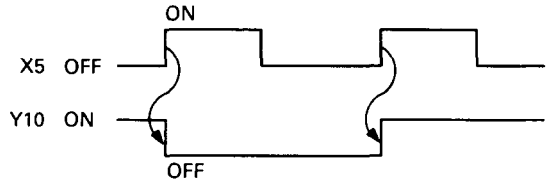
(D1)	Required device number
(D2)	Dummy data Any device number indicated by △

Functions

(1) Reverses the output status of the device, (D1), on the leading edge of the output reverse command.

(2) Specify any device number indicated by (dummy data).



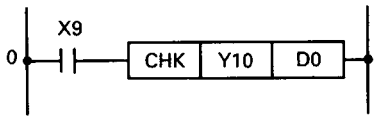


(3) The output reverse command on/off period must be equal or greater than 1 scan time.

(4) The CHK instruction is only executed in refresh mode.

Program Example

The following program reverses the output status of Y10 when X9 is switched on.



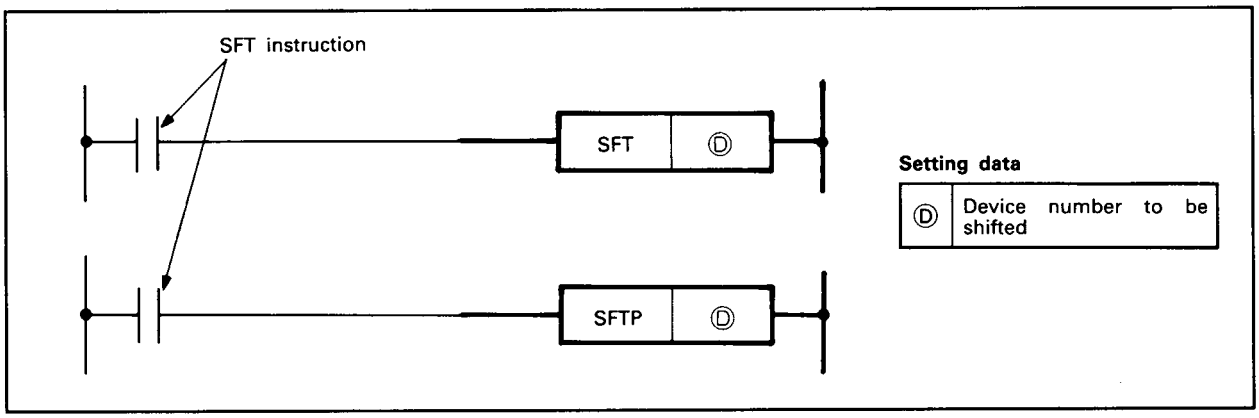
Coding

Step Number	Instruction	Device			
0	LD	X9			
1	CHK	Y10	D0		
6	END				

5.4 Shift Instructions
5.4.1 Bit device shift (SFT, SFTP)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○																3					

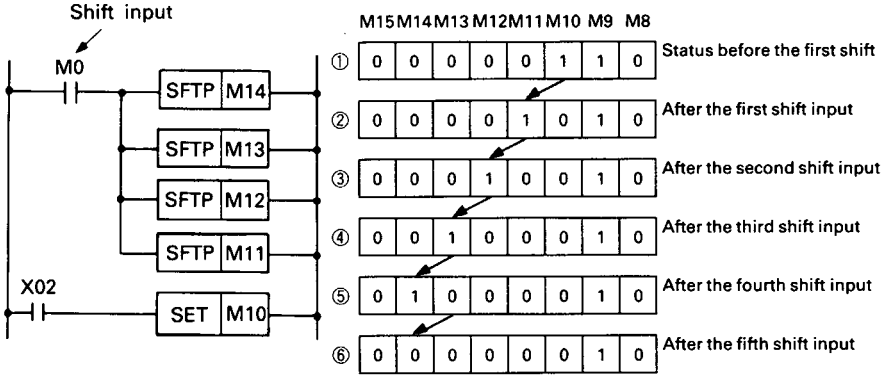


Functions

(1) This instruction shifts the ON/OFF status of a device number, (defined as D-1) to the device specified as D and turns off the device with the lower number.

(2) Turn on the head device to be shifted with the SET instruction.

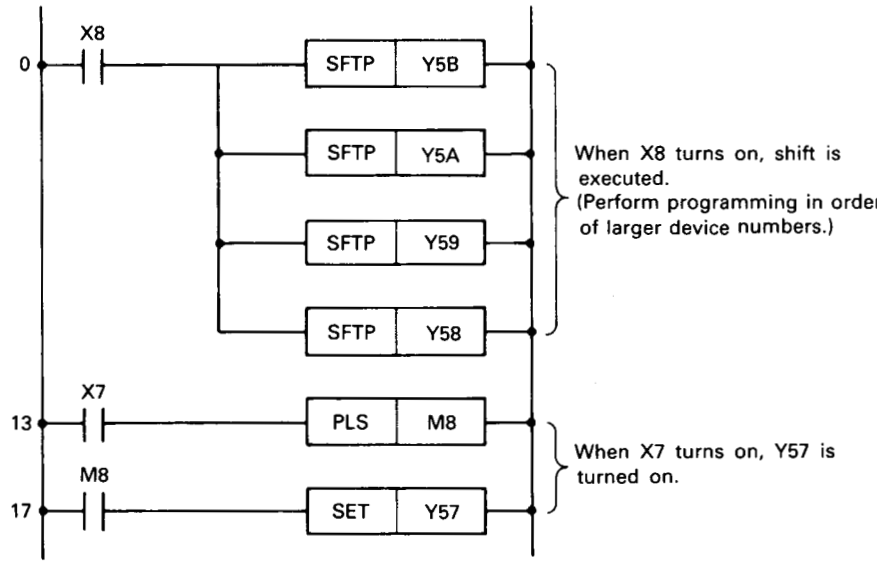
(3) When the SFT or SFTP instruction is used consecutively, program higher device numbers first. (See below.)

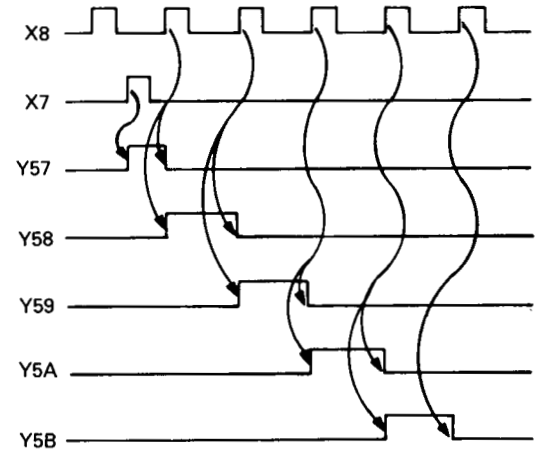


*: At M8 to 15, 1 indicates ON and 0 indicates OFF.

Program Example

1) Program which shifts the Y57 to 5B when X8 turns on.





Coding

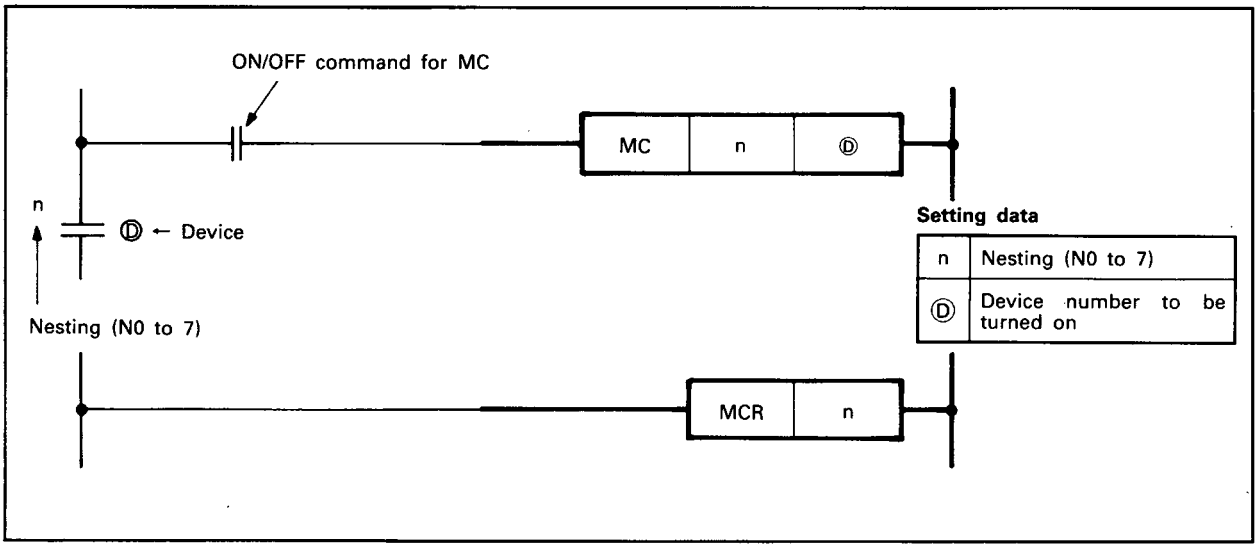
Step Number	Instruction	Device			
0	LD	X8			
1	SFTP	Y5B			
4	SFTP	Y5A			
7	SFTP	Y59			
10	SFTP	Y58			
13	LD	X7			
14	PLS	M8			
17	LD	M8			
18	SET	Y57			
19	END				

5.5 Master Control Instructions

5.5.1 Master control set, reset (MC, MCR)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n																					○		5/3					
Ⓓ		○	○	○	○	○	○																					



Functions

MC

(1) MC is master control start instruction. when the ON/OFF command for the MC is on, operation results from MC to MCR remain unchanged.

(2) When ON/OFF command for the MC is off, the operation result of MC to MCR is as indicated below.

100ms, 10ms Timers	100ms integrating Timer Counter	OUT Instruction	SET/RST	SFT
Count value turns to 0.	Remain as a present count value.	All turn to 0.	Status is held.	

(3) Nesting is allowed up to eight levels (N0 to 7). Nest MC starting with lower nesting numbers (N), and nest MCR starting with higher numbers. For the nesting, refer to Section 2.4.12.

(4) Scanning of the program between the MC and MCR instructions occurs constantly, even when the MC is de-energized.

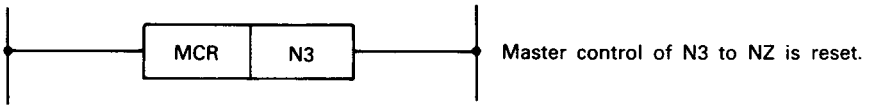
(5) The MC instruction can be used any number of times during one scan by changing the device of destination D.

(6) When the MC instruction is on, the coil of device specified at the destination turns on. The use of the same device for an OUT instruction, etc. is trated as a duplicate coil.

MCR

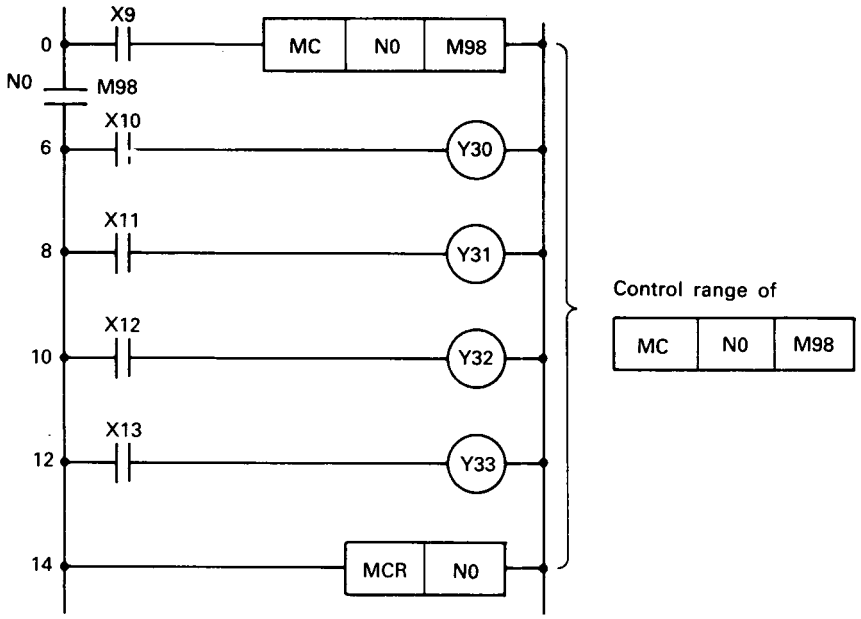
(1) MCR is a master control reset instruction and indicates the end of master control range.

(2) MCR resets the specified nesting (N) number and succeeding numbers.



Program Example

1) Program which turns on MC when X9 turns on and which turns off MC when X9 turns off.



Coding

Step Number	Instruction	Device			
0	LD	X9			
1	MC	N0	M98		
6	LD	X10			
7	OUT	Y30			
8	LD	X11			
9	OUT	Y31			
10	LD	X12			
11	OUT	Y32			
12	LD	X13			
13	OUT	Y33			
14	MCR	N0			
17	END				

REMARKS

The numbers of steps required for the MC and MCR instructions are as follows:

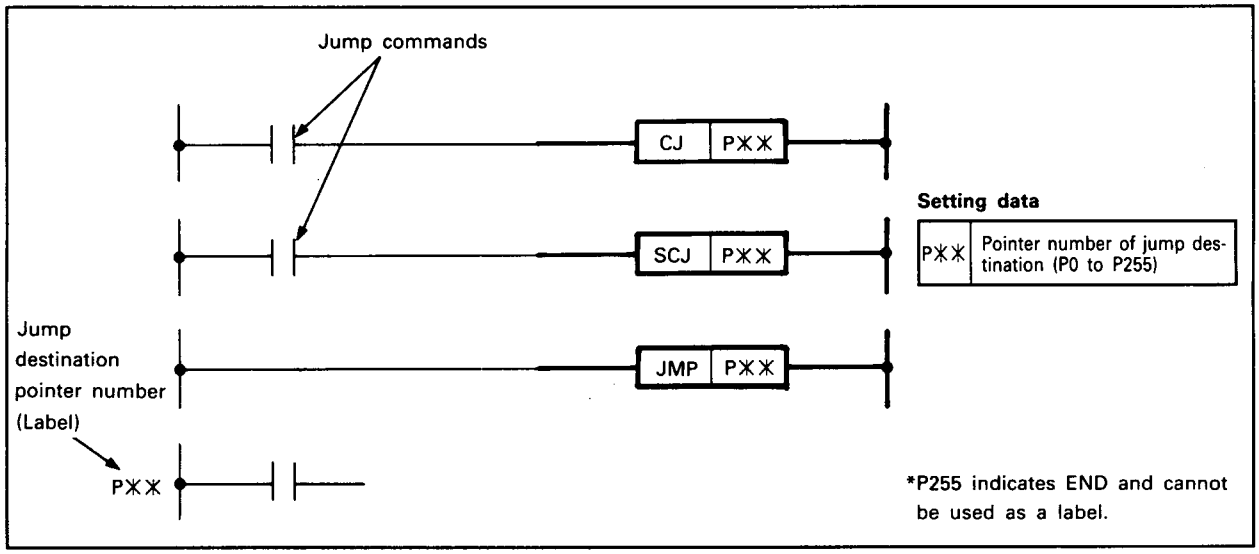
MC: 5 steps
MCR: 3 steps

5.6 Program Branch Instructions

5.6.1 Conditional jump, unconditional jump (CJ, SCJ, JMP)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
P																			○				3		○		○	○

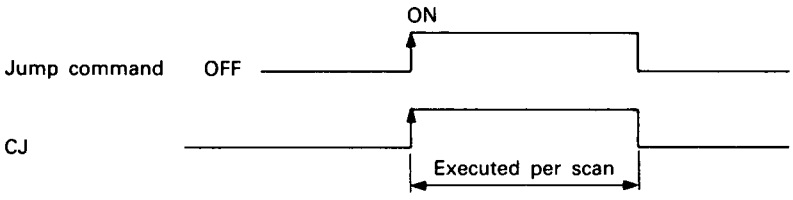


Functions

CJ

(1) Executes the program of specified pointer number when the jump command is on.

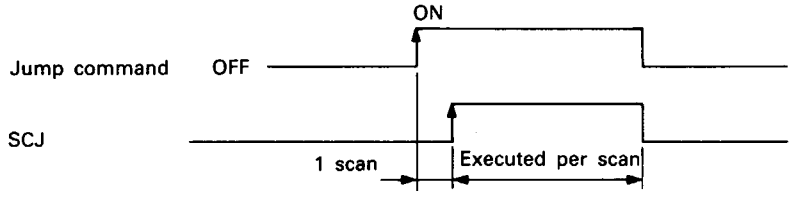
(2) Executes the program of the next step when the jump command is off.



SCJ

(1) Executes the program of specified pointer number, starting at the next scan, when the jump command changes from off to on.

(2) Executes the program of the next step when the jump command is off or changes from off to on.



JMP

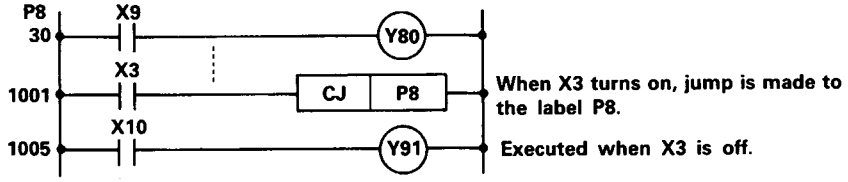
(1) Executes the program of specified pointer number unconditionally.

POINT

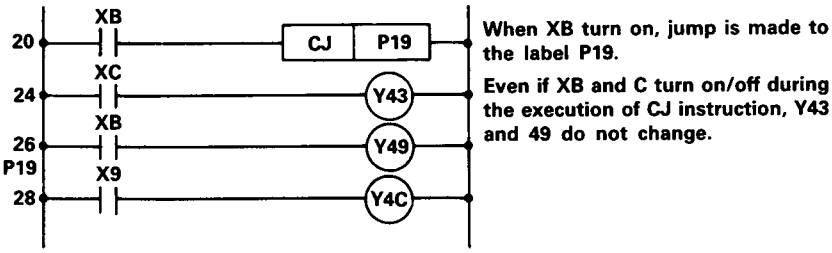
(1) Even if the timer, of which coil is on, is jumped by the CJ, SCJ, or JMP instruction after the coil of timer is turned on, the timer continues counting.

(2) When a jump is made to a memory location by CJ, SCJ, or JMP, the scan timer is shortened.

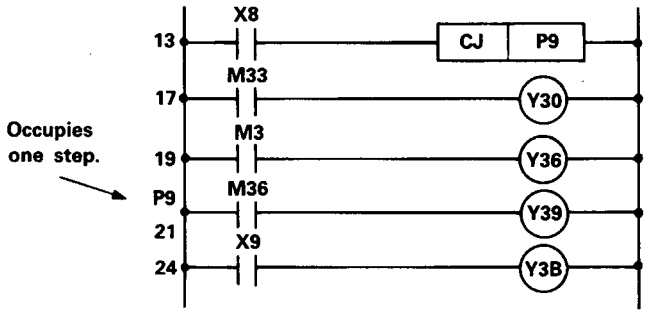
(3) The CJ, SCJ, and JMP instructions are also capable of jumping to a step with lower number. However, when utilizing this capability, execute the WDT instruction or the END (FEND) instruction before the watch dog timer times out.



(4) The device jumped by CJ, SCJ, or JMP does not change.



(5) The label (P✳✳) occupies one step.



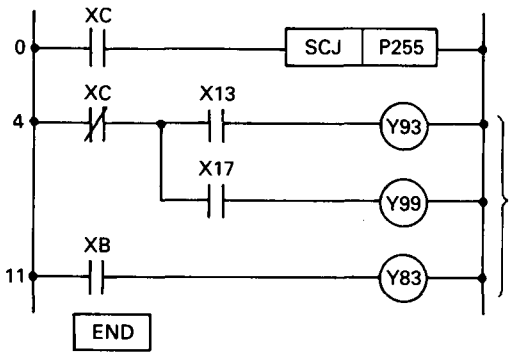
Grammatical Errors

In the following cases, grammatical error occurs and the PC stops its operation.

- When there are mult. contacts of the same labels, a jump has been made to that label by the CJ, SCJ, or JMP instruction.
- There is no label at the jump destination of CJ, SCJ, or JMP instruction.
- Jump has been made to a label located below the END instruction.

Program Examples

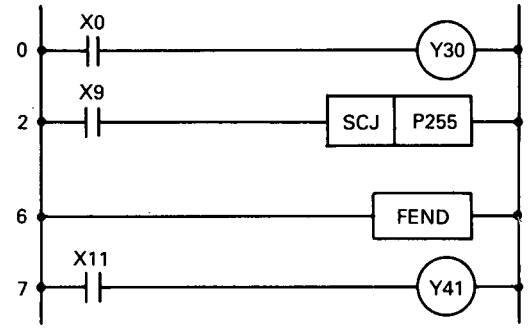
1) Program which causes a jump during the next scan to END when XC turns on.



Coding

Step Number	Instruction	Device			
0	LD	XC			
1	SCJ	P255			
4	LDI	XC			
5	MPS				
6	AND	X13			
7	OUT	X93			
8	MPP				
9	AND	X17			
10	OUT	Y99			
11	LD	XB			
12	OUT	Y83			
13	END				

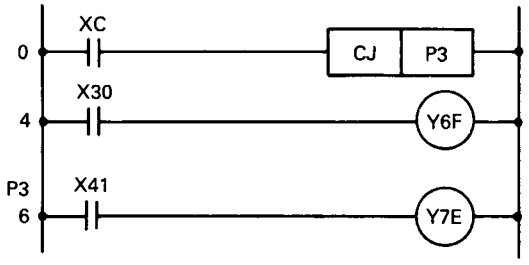
2) Program which causes a jump to the END (FEND) instruction when X9 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	Y30			
2	LD	X9			
3	SCJ	P255			
6	FEND				
7	LD	X11			
8	OUT	Y41			
9	END				

3) Program which causes a jump during the next scan to P3 when XC turns on.



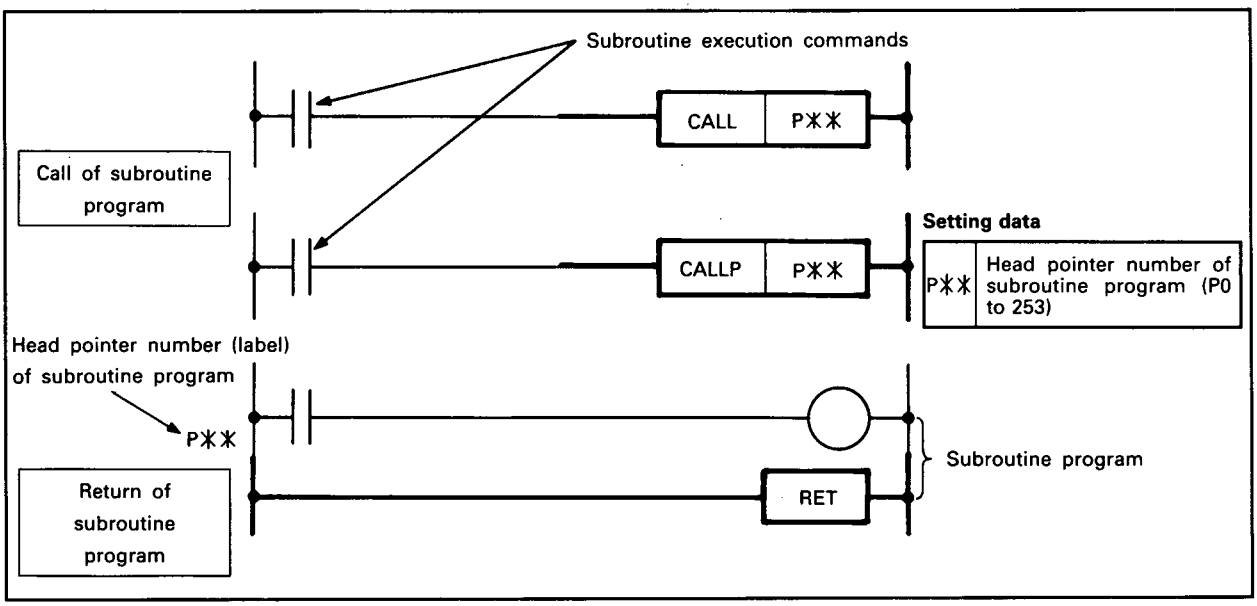
Coding

Step Number	Instruction	Device			
0	LD	XC			
1	CJ	P3			
4	LD	X30			
5	OUT	Y6F			
6	P3				
7	LD	X41			
8	OUT	Y7E			
9	END				

5.6.2 Subroutine call, return (CALL, CALLP, RET)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
P																			○				3/1		○		○	○

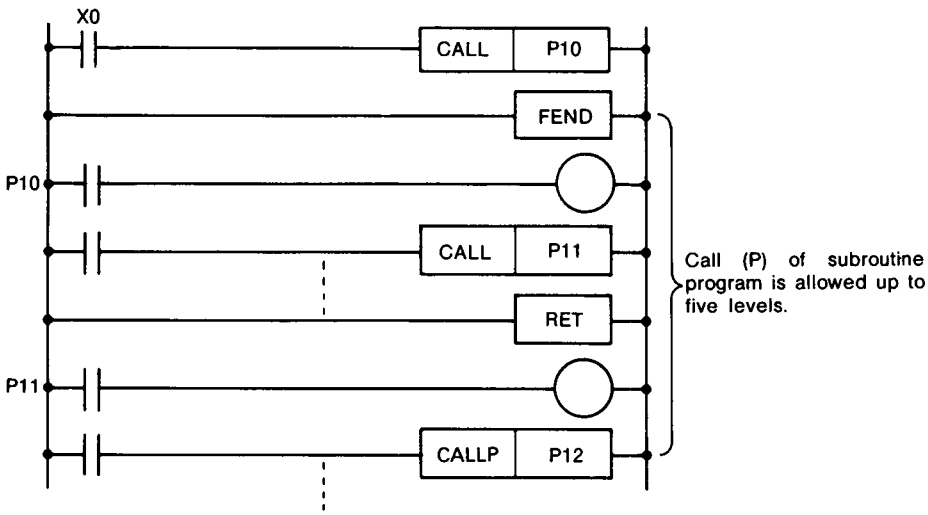


Functions

CALL, CALLP

(1) Executes the subroutine program specified by the pointer (P**).

(2) Up to five levels of nesting of the CALL/CALLP instruction are allowed.



(3) The NOP instruction must be written between the RET instruction in the subroutine program and the END instruction in the sequence program.

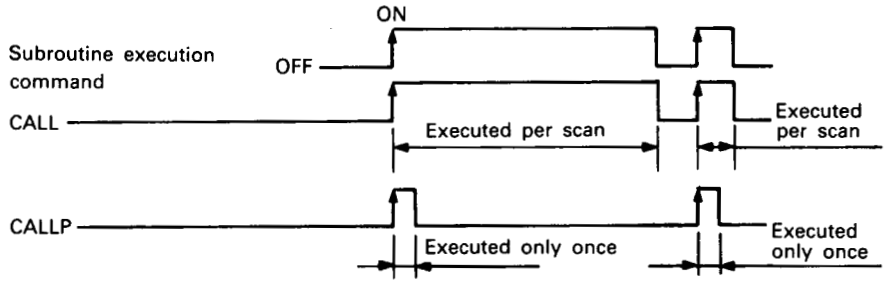
RET

(1) Executes the sequence program located at the next step to the CALL(P) instruction when the RET instruction is executed.

(2) Indicates the end of subroutine program.

Execution Conditions

The execution conditions of CALL and CALLP are a shown below.



Grammatical Errors

In the following cases, grammatical error occurs and the PC stops operation.

- After the CALL(P) instruction is executed, the END (FEND) instruction has been executed before executing the RET instruction.
- The RET instruction has been executed before executing the CALL(P) instruction.
- The label P255 has been called by the CALL(P) instruction.

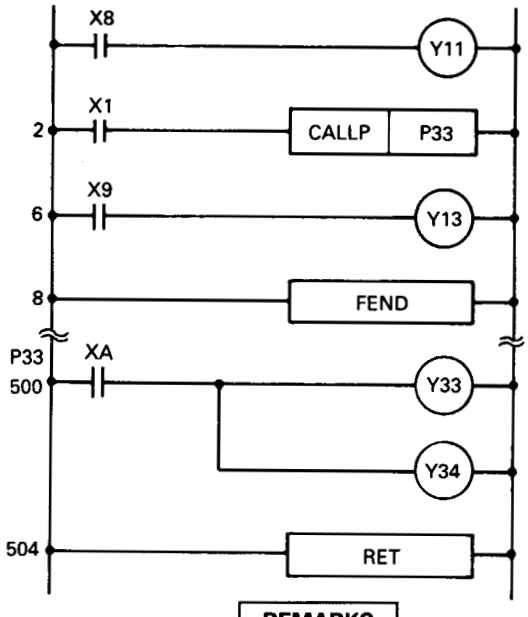
Operation Error

In the following case, operation error occurs and the error flag turns on.

- Nesting is of six or more levels.

Program Example

1) Program which executes the subroutine program when X1 changes from off to on.



Coding

Step Number	Instruction	Device			
0	LD	X8			
1	OUT	Y11			
2	LD	X1			
3	CALLP	P33			
6	LD	X9			
7	OUT	Y13			
8	FEND				
		⟆			
500	P33				
501	LD	XA			
502	OUT	Y33			
503	OUT	Y34			
504	RET				
505	END				

REMARKS

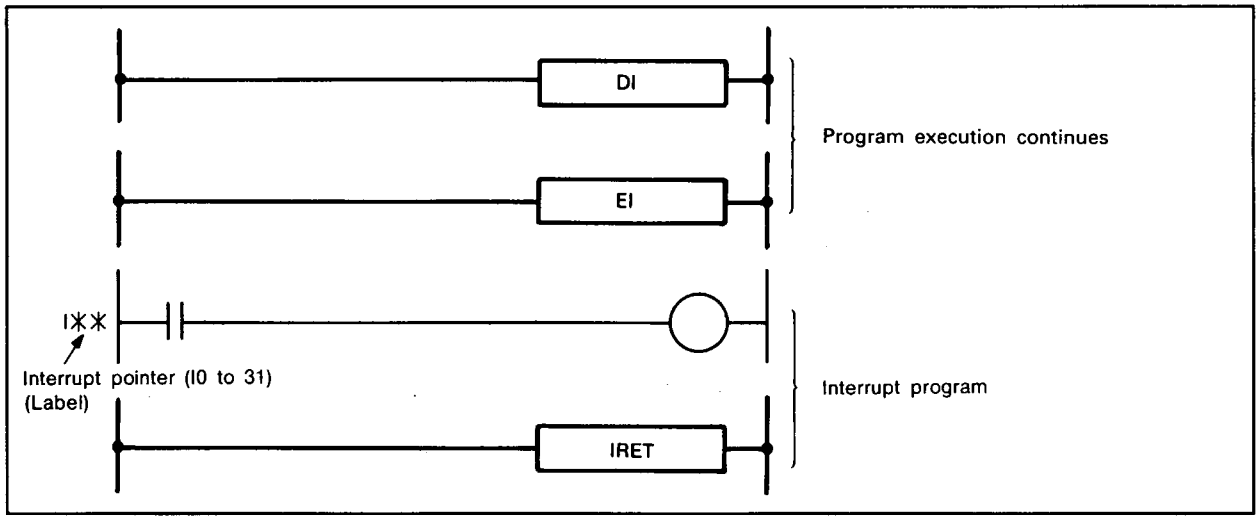
The numbers of steps of the CALL(P) and RET instructions are as follows:

CALL(P): 3 steps
RET: 1 step

5.6.3 Interrupt enable, disable, return (EI, DI, IRET) (Valid with M9053 off when the A□NCPU is used)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

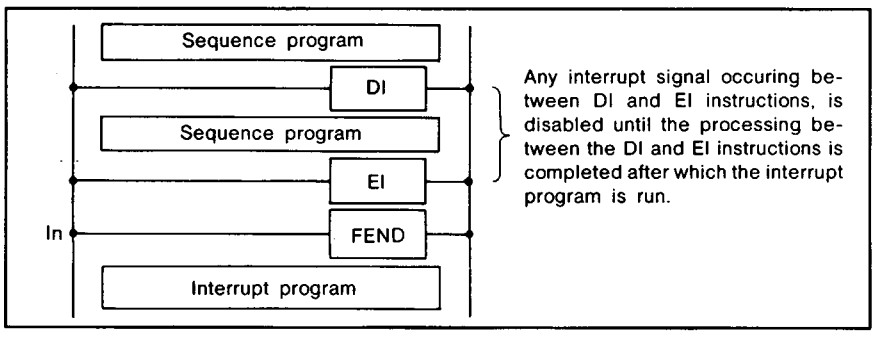
DI

(1) Disables the interrupt program until the EI instruction is executed so that interrupt signals are ignored.
Valid with M9053 off when the A□NCPU is used.

(2) When the PC CPU is RESET, interrupt program execution is disabled.

EI

(1) Enables the interrupt program.
Valid with M9053 off when the A□NCPU is used.



IRET

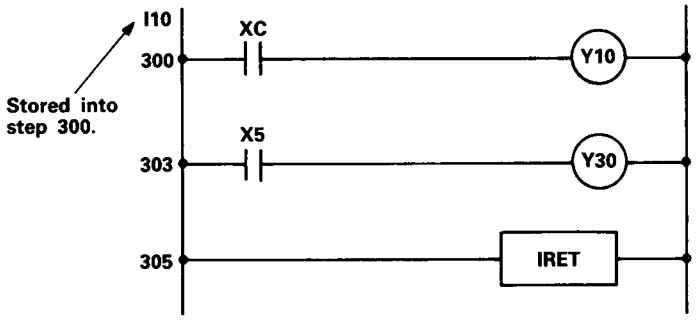
(1) Indicates the termination of processing of interrupt program.

(2) Performs the processing of counter for interruption and returns the processing to the sequence program after the RET instruction is executed.

POINT

(1) When a counter is used in the interrupt program, use the counter for interruption.
The A3HCPU does not have any counter which may be used in the interrupt program.

(2) The pointer for interruption occupies one step.

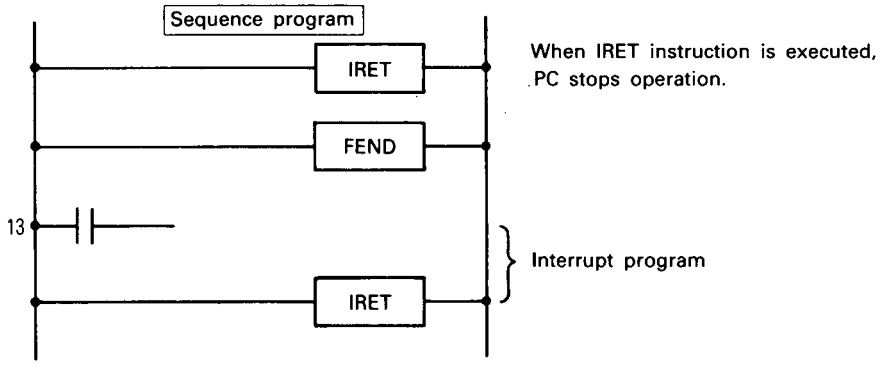


(3) For the interrupt conditions, refer to Section 2.4.14.

(4) During the execution of interrupt program, DI (interruption inhibition) is set. Do not allow multiple interrupt programs to be run simultaneously. This can be prevented by using the EI instruction in the interrupt programs.

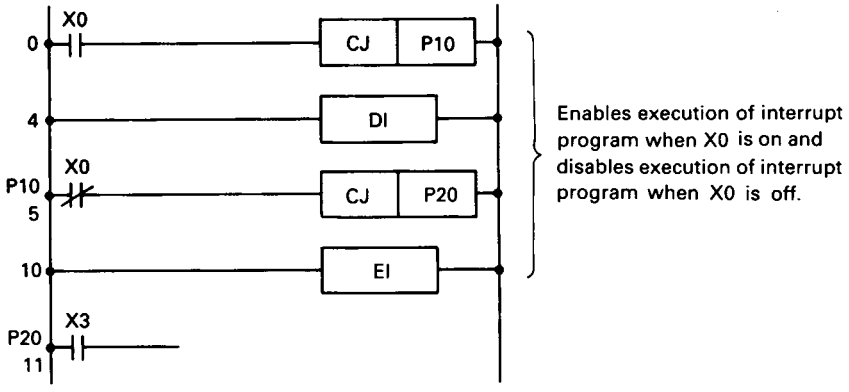
Grammatical Error

If the IRET instruction is executed prior to the run of interrupt program, the PC stops its operation.



Program Example

Disable/enable program of the run of interrupt program by DI and EI.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	CJ	P10			
4	DI				
5	P10				
6	LDI	X0			
7	CJ	P20			
10	EI				
11	P20				
12	LD	X3			
		⟆			

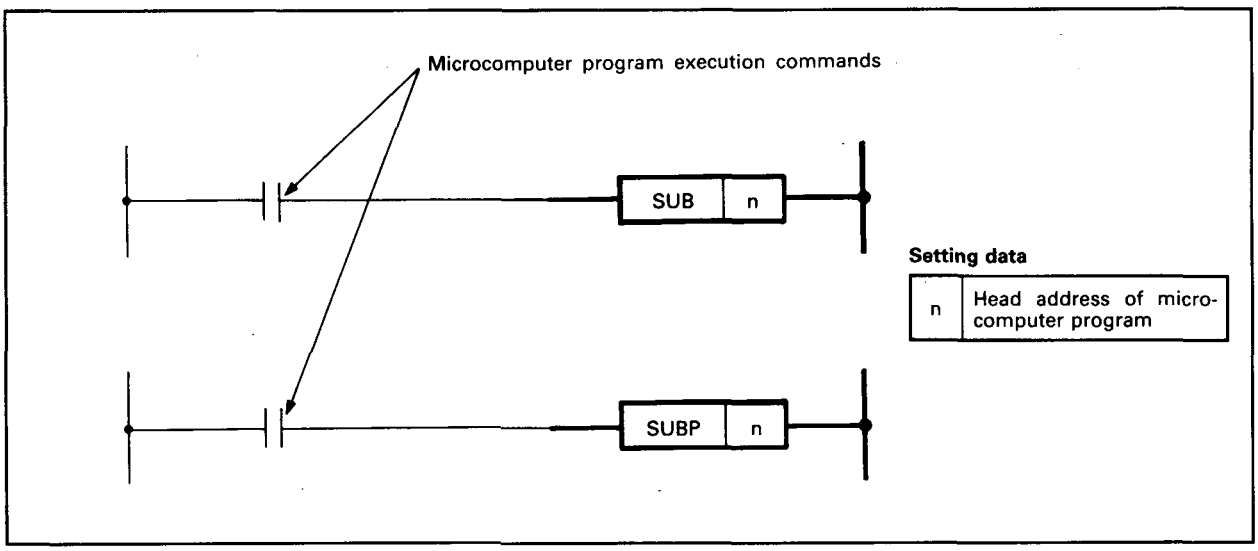
5

MEMO

5.6.4 Microcomputer program call (SUB, SUBP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n								○	○	○	○	○	○	○	○	○	○	○					3		○		○	○



Setting data

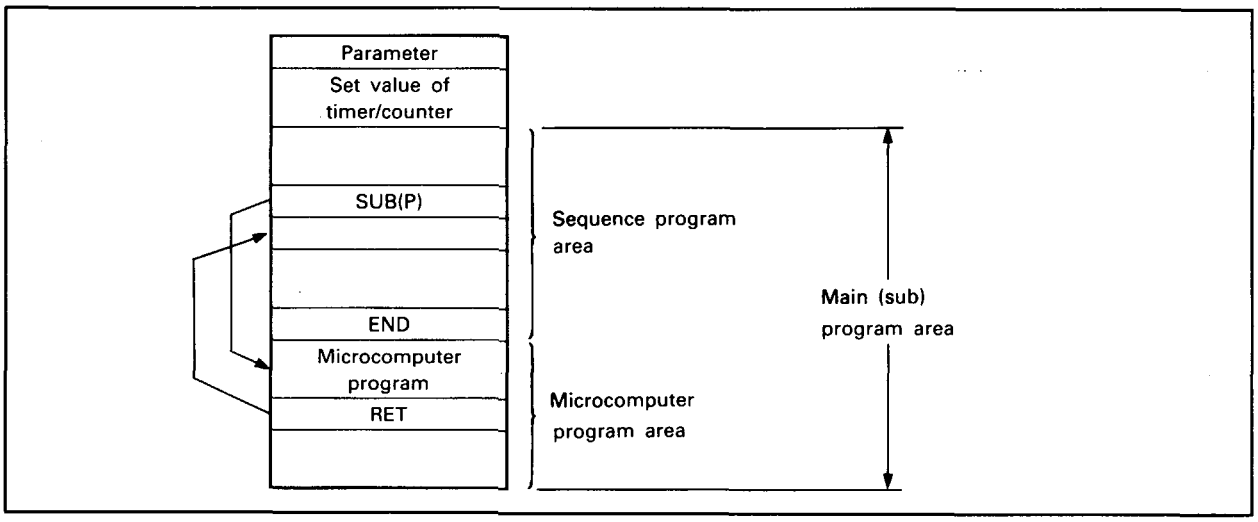
n	Head address of micro-computer program

Functions

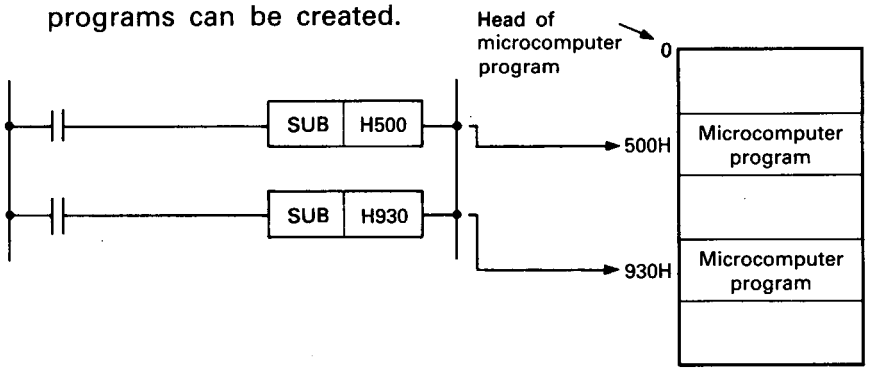
(1) Calls the microcomputer program created by user and allows the run of microcomputer program.

(2) When the run of microcomputer program is completed, runs the sequence program again, starting at the next step to the SUB or SUBP instruction.

(3) The SUB and SUBP instructions can be used for the sequence program and subsequence program.



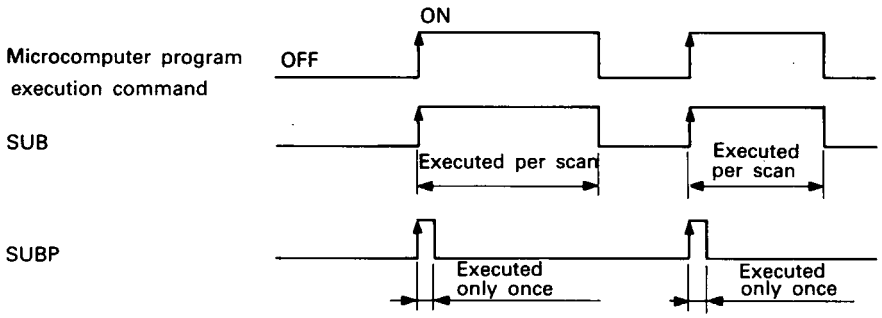
(4) In the microcomputer program area, multiple microcomputer programs can be created.



(5) For the details of microcomputer program, refer to Chapter 8.

Execution Conditions

The execution conditions of SUB and SUBP instructions are as shown below.



Operation Error

In the following case, operation error occurs and the error flag turns on.

- An area of more than the microcomputer program capacity has been specified at n.

5

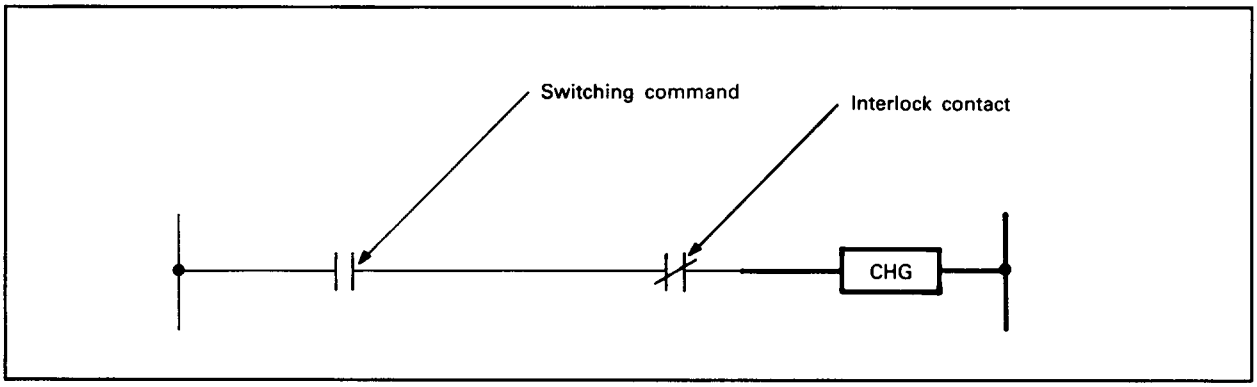
IB (NA) 66147-A

5.7 Program Switching Instructions

5.7.1 Main ←→subprogram switching (CHG)

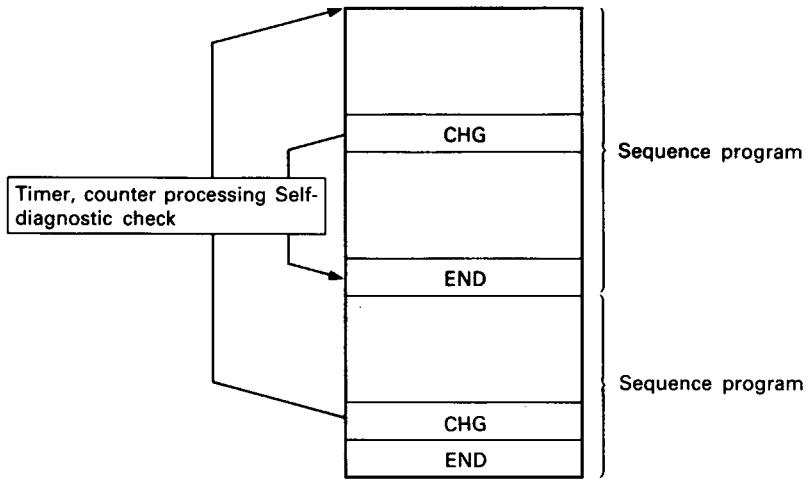
Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N				M9012	M9010	M9011
																						1				



Functions

(1) Executes switching between the main program and subprogram after the timer/counter processing and self-diagnostic check.

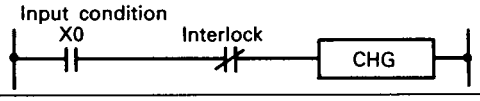
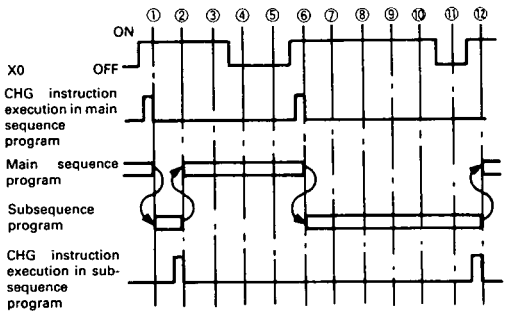


(2) For further information, see Section 3.8.

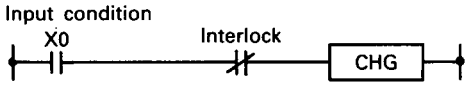
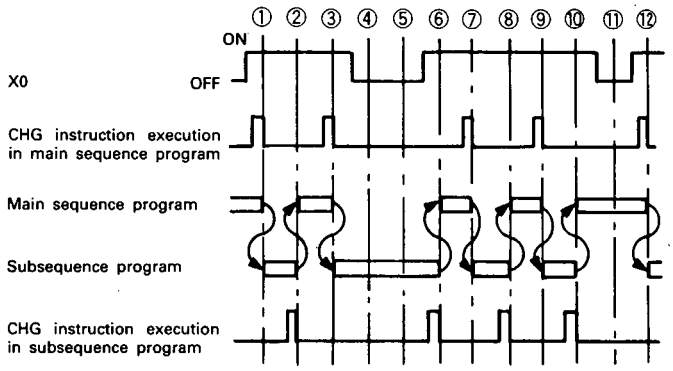
5

Execution Conditions

(1) When the A3NCPU is used, the CHG instruction is only executed on the leading edge of its input condition. (There is no M9050.)

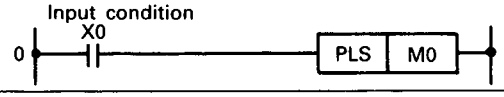
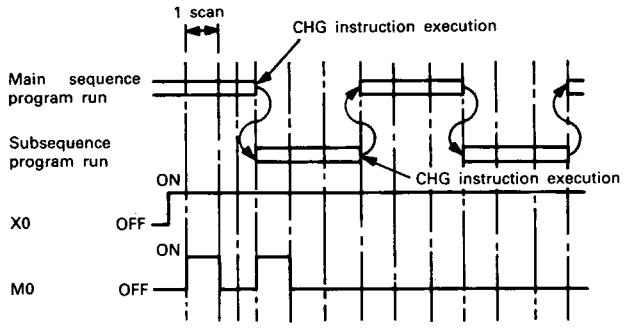
Ladder example		The following program is written before END or FEND of the main and subsequence programs. 
Timing chart		
Operation depending on ON/OFF of X0	OFF	No switching between the main and subsequence programs. (④, ⑤, ⑪)
	ON	The main sequence program is only switched to the subsequence program, then back to the main sequence program on the first leading edge of the CHG instruction execution command (X0). (②)
	OFF ↓ ON	Switched between the main and subsequence programs (①, ⑥, ⑫)
Remarks		When the CHG instruction is executed, END processing (e.g. timer timing, counter counting, WDT reset) is performed for the current program and operation is started from step 0 of the other program.

(2) When the A3HCPU is used, the CHG instruction is executed repeatedly while its input condition is on.

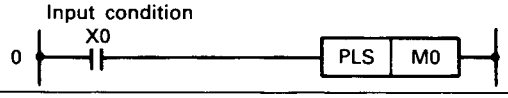
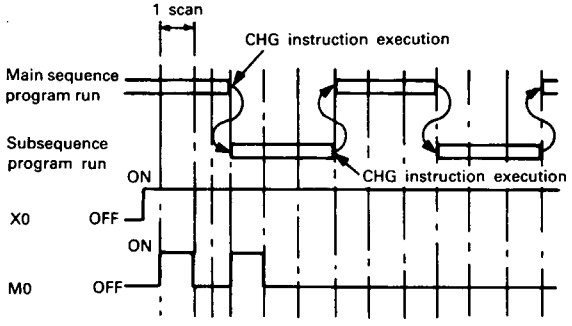
Ladder example		The following program is written before END or FEND of the main and subsequence programs. 
Timing chart		
Operation depending on ON/OFF of X0	OFF	No switching between the main and subsequence programs. (④, ⑤, ⑪)
	ON	CHG instruction is executed every scan and switches between the main and subsequence programs. (②, ③, ⑦, ⑧, ⑨, ⑩)
	OFF ↓ ON	Switched between the main and subsequence programs (①, ⑥, ⑫)
Remarks		When the CHG instruction is executed, END processing (e.g. timer timing, counter counting, WDT reset) is performed for the current program and operation is started from step 0 of the other program.

Execution of PLS Instruction Used with CHG Instruction

(1) When the A3NCPU is used, the PLS instruction is executed as indicated below.

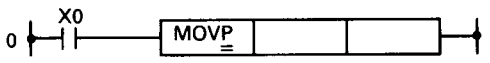
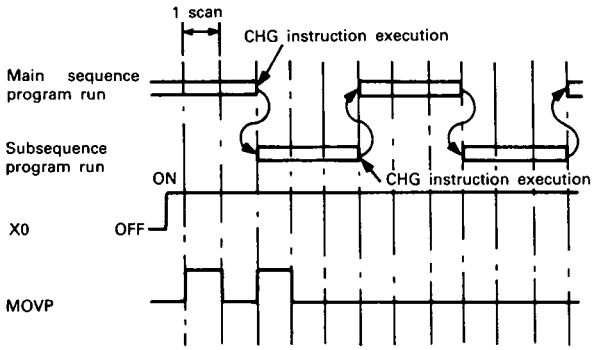
Ladder example			The following program is written at step 0 of the main and subsequence programs. 
Timing chart			
Operation	X0 status	OFF	M0 is not switched on.
		ON	M0 is only switched on during the first scan of the subsequence program selected by the CHG instruction executed after X0 is switched on.
		OFF ↓ ON	M0 is only switched on during 1 scan.

(2) When the A3HCPU is used, the PLS instruction is executed as indicated below.

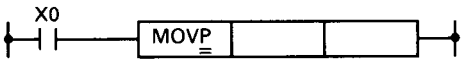
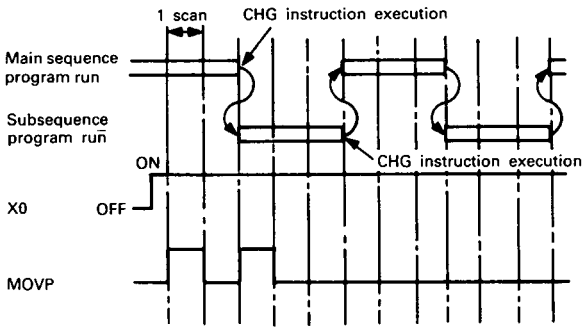
Ladder example			The following program is written at step 0 of the main and subsequence programs. 
Timing chart			
Operation	X0 status	OFF	M0 is not switched on.
		ON	M0 is only switched on during the first scan of the subsequence program selected by the CHG instruction executed after X0 is switched on.
		OFF ↓ ON	M0 is only switched on during 1 scan.

Execution of ⬚ P Instruction Used with CHG Instruction

(1) When the A3NCPU is used, the MOVP instruction is executed as indicated below.

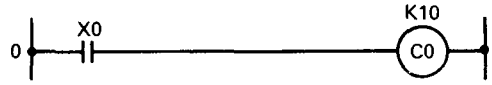
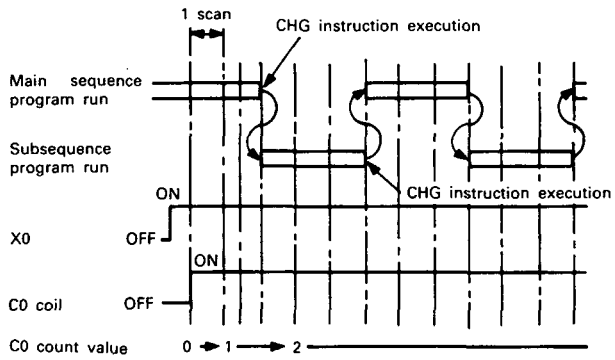
Ladder example		The following program is written at step 0 of the main and subsequence programs. 
Timing chart		
Operation depending on X0 ON/OFF status	OFF	MOVP instruction is not executed.
	ON	MOVP instruction is only executed during the first scan of the subsequence program selected by the CHG instruction executed after X0 is switched on.
	OFF ↓ ON	MOVP instruction is only executed once.

(2) When the A3HCPU is used, the MOVP instruction is executed as indicated below.

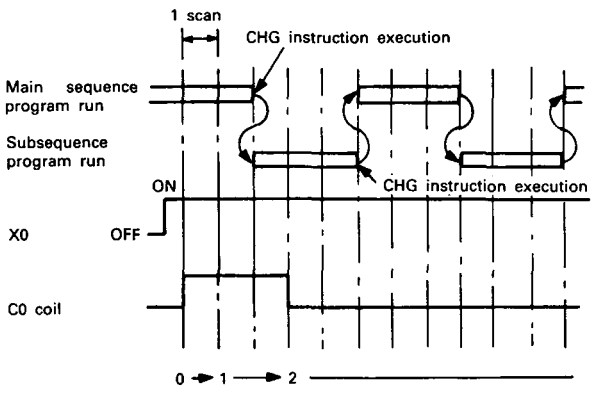
Ladder example		The following program is written at step 0 of the main and subsequence programs. 
Timing chart		
Operation depending on X0 ON/OFF status	OFF	MOVP instruction is not executed.
	ON	MOVP instruction is only executed during the first scan of the subsequence program selected by the CHG instruction executed after X0 is switched on.
	OFF ↓ ON	MOVP instruction is only executed once.

Counting of Counter Used with CHG Instruction

(1) When the A3NCPU is used, the counter counts as indicated below.

Ladder example		The following program is written at step 0 of the main and subsequence programs. 
Timing chart		
Operation depending on X0 ON/OFF status	OFF	C0 count value remains unchanged.
	ON	C0 count value is incremented by 1 after END (FEND, CHG) is executed during the first scan of the program selected by the CHG instruction executed after X0 is switched on.
	OFF ↓ ON	C0 count value is incremented by 1 after END (FEND, CHG) is executed.

(2) When the A3HCPU is used, the counter counts as indicated below.

Ladder example		The following program is written at step 0 of the main and subsequence programs. 
Timing chart		
Operation depending on X0 ON/OFF status	OFF	C0 count value remains unchanged.
	ON	C0 count value is incremented by 1 after END (FEND, CHG) is executed during the first scan of the program selected by the CHG instruction executed after X0 is switched on.
	OFF ↓ ON	C0 count value is incremented by 1 after END (FEND, CHG) is executed.

IB (NA) 66147-A

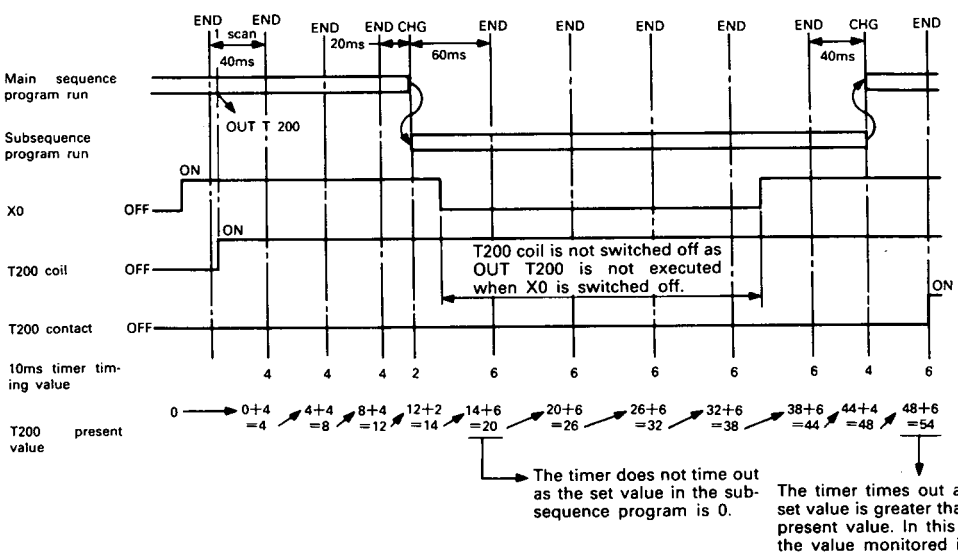
Timing of Timer Used with CHG Instruction

The A3N and A3HCPUs each have two timer set value storage areas; one for the main sequence program and the other for the subsequence program.
In these areas, the set value of the timer not in use is 0. The set value of 0 is regarded as infinite and the timer does not time out.
When the main (sub) sequence program is switched to the sub (main) sequence program by the CHG instruction after the timer in the main (sub) sequence program has started timing, the timer does not time out during execution of the sub (main) program because the timer set value specified in the main (sub) program is 0 in the sub (main) program timer set value storage area.

Ladder example

The following program is written after the main sequence program and the same timer number is not used in the subsequence program.

Timing chart

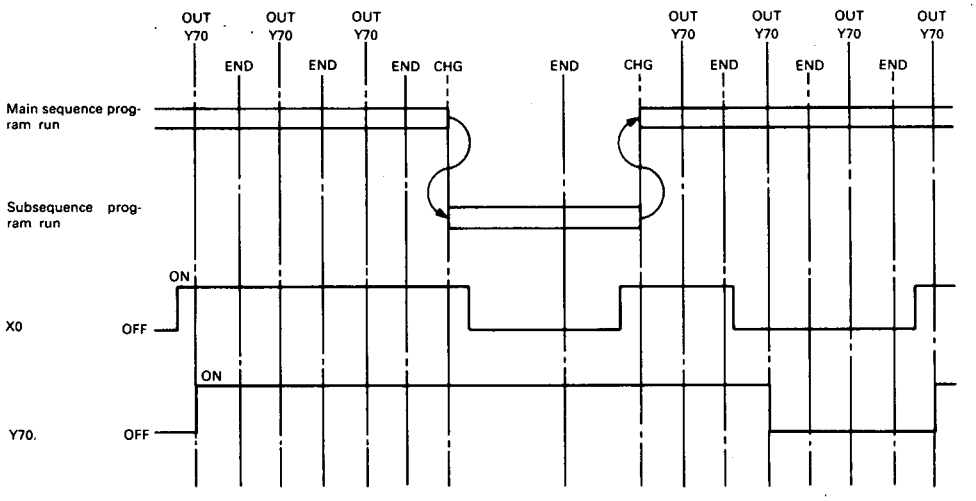


Operation

T200 started by the main sequence program does not time out while the subsequence program is running.
It times out on the following condition when the main sequence program is run again:
(Present value) < 0 or (set value) < (present value)

Execution of OUT Instruction Used with CHG Instruction

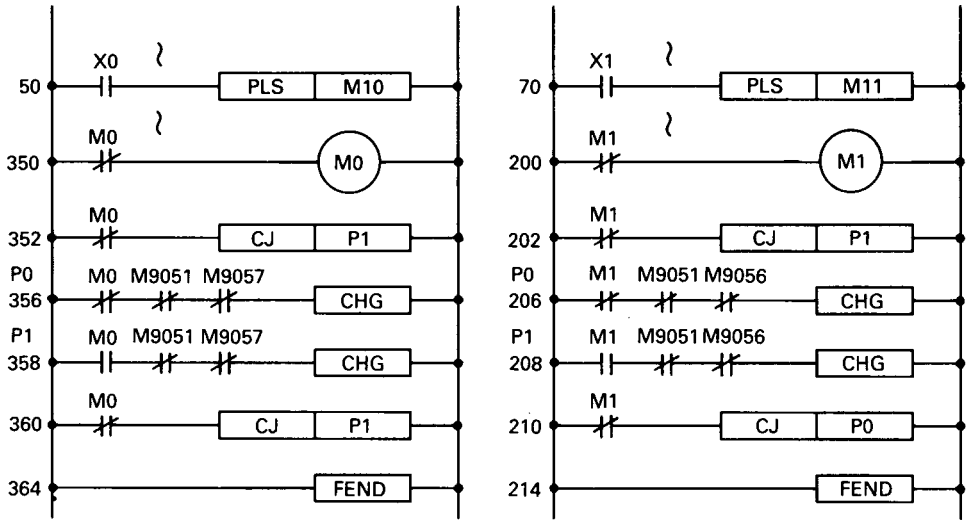
When the A3N, A3HCPU is used, the coil switched on/off in the main (sub) sequence program remains unchanged during sub (main) sequence program run if its input condition changes.

Ladder example	The following program is written after the main sequence program and the same coil is not used in the subsequence program. 
Timing chart	
Operation	Y70 is switched on/off when X0 is switched on/off during main sequence program run. Y70 remains unchanged if X0 is switched on/off during subsequence program run.

Program Examples

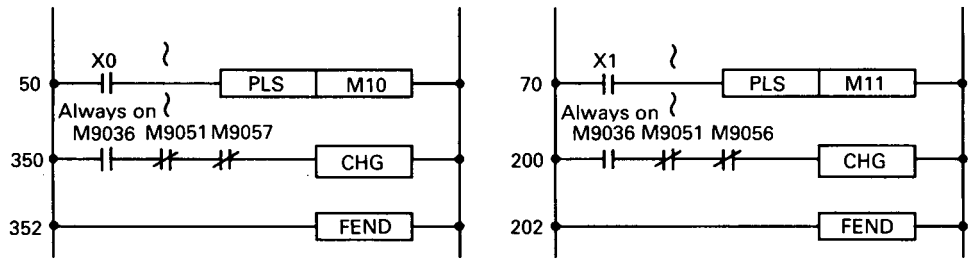
The following programs are used with the A3N and A3HCPUs to output pulses in accordance with the input condition of the PLS instruction while alternately running the main and subprograms.

(1) A3NCPU program



Main sequence program　　Subsequence program

(2) A3HCPU program



Main sequence program　　Subsequence program

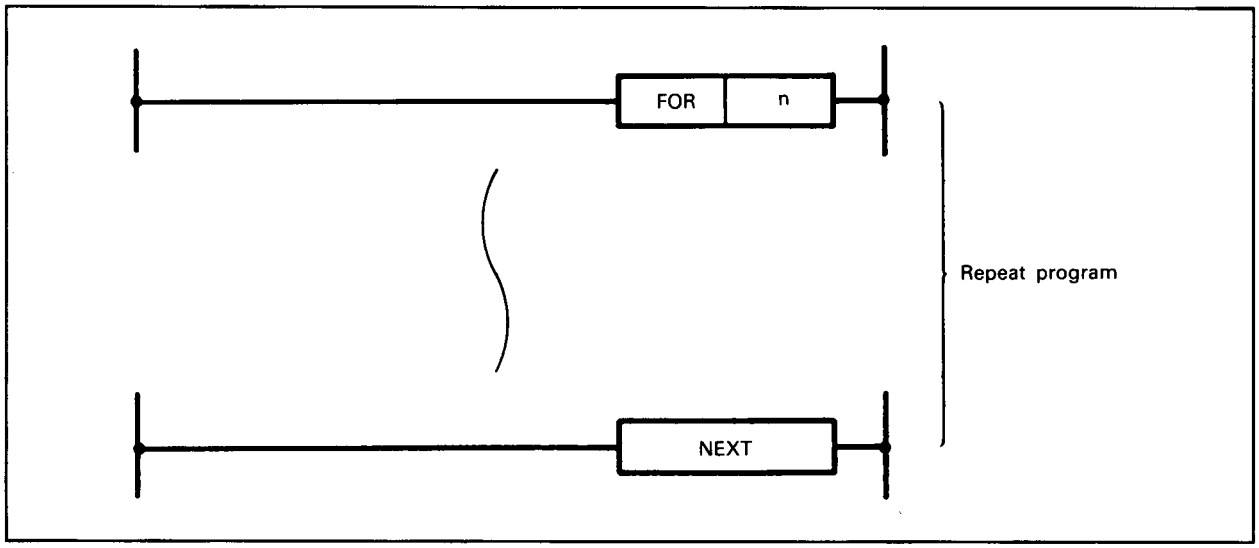
CAUTION

When modifying a subprogram during main program run or vice versa, M9051, M9056 and M9057 contacts should be used to disable the CHG instruction so that the CHG instruction may not switch the currently running program to the program currently being corrected. For further details, see Section 3.8.4.

5.8 FOR ~ NEXT Instructions
5.8.1 FOR ~ NEXT (FOR, NEXT)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n								○	○	○	○	○	○	○	○	○	○	○					3/1				○	○



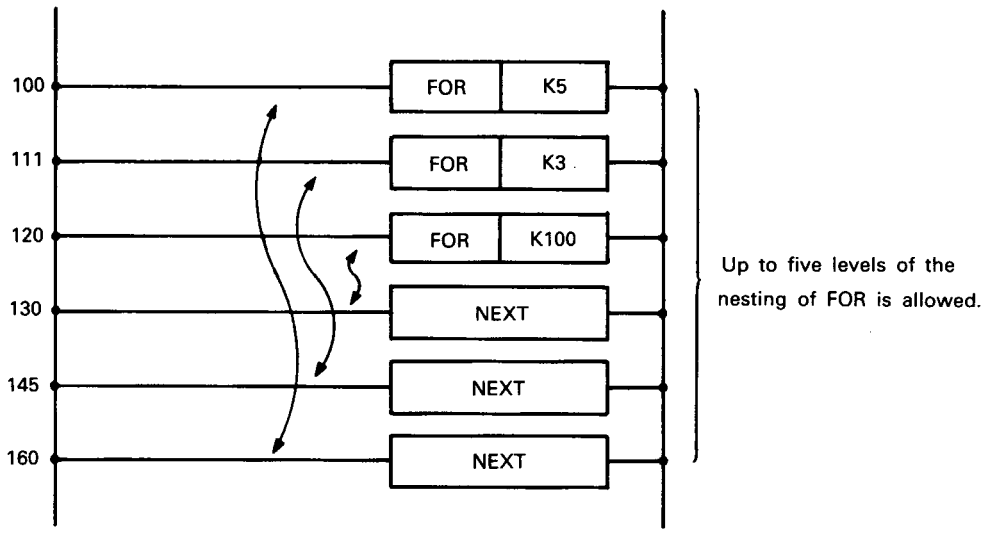
Functions

(1) When the processing of FOR to NEXT instructions is executed "n" times unconditionally, performs the processing of the next step to the NEXT instruction.

(2) At "n", 1 to 32767 can be specified. When −32767 to 0 has been specified, the same processing an n=1 is performed. (positive integers)

(3) When it is not desired to execute the processing of FOR to NEXT instructions, cause a jump by use of the CJ or SCJ instruction.

(4) Up to five levels of the nesting of FOR is allowed.



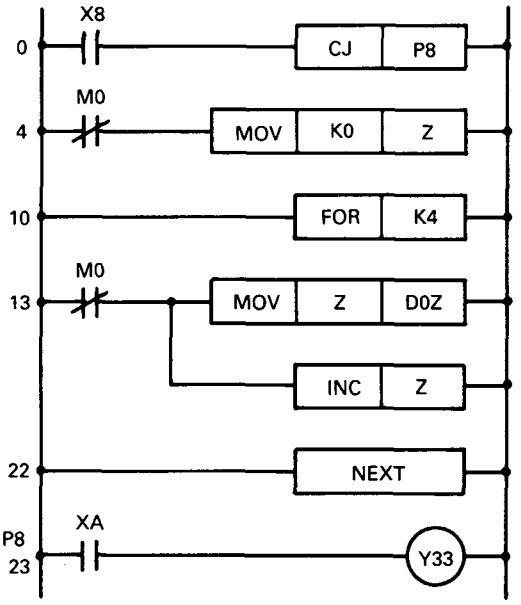
Grammatical Errors

In the following cases, grammatical error occurs and the PC stops its operation.

- ○ After the execution of FOR instruction, the END (FEND) instruction has been executed before the NEXT instruction is executed.
- ○ The NEXT instruction has been executed before the FOR instruction is executed.

Program Example

1) Program which executes the FOR to NEXT instructions when X8 is off and does not execute the FOR to NEXT instructions when X8 is on.



Coding

Step Number	Instruction	Device			
0	LD	X8			
1	CJ	P8			
4	LDI	M0			
5	MOV	K0	Z		
10	FOR	K4			
13	LDI	M0			
14	MOV	Z	D0Z		
19	INC	Z			
22	NEXT				
23	P8				
24	LD	XA			
25	OUT	Y33			
26	END				

REMARKS

The numbers of steps of the FOR and NEXT instructions are as follows:

- FOR: 3 steps
- NEXT: 1 step

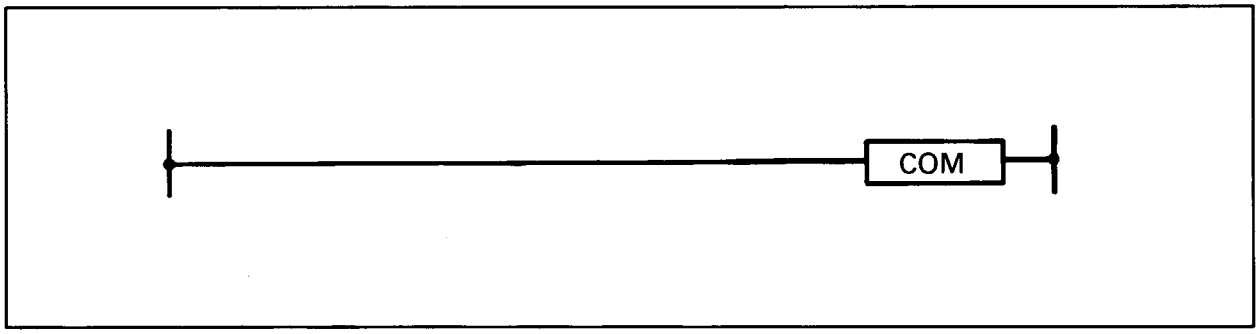
5

5.9 Link Refresh Instructions

5.9.1 Link refresh (COM)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

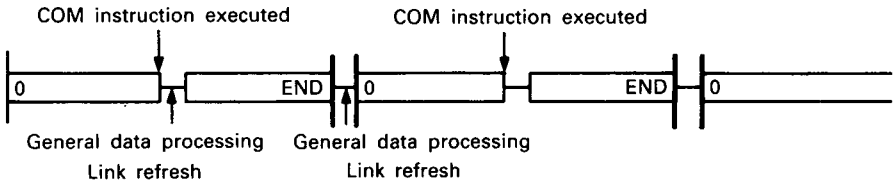
Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						3					



Functions

(1) The COM instruction is used to make faster data communication with a remote I/O station or to receive data positively when the scan time of the master station sequence program is longer than that of the local station sequence program.

(2) On execution of the COM instruction, the PC CPU temporarily stops the sequence program processing and performs general data processing (END processing) and link refresh processing.



(3) The COM instruction may be used any number of times in the sequence program. In this case, note that the sequence program scan time increases the period of general data processing and link refresh times.

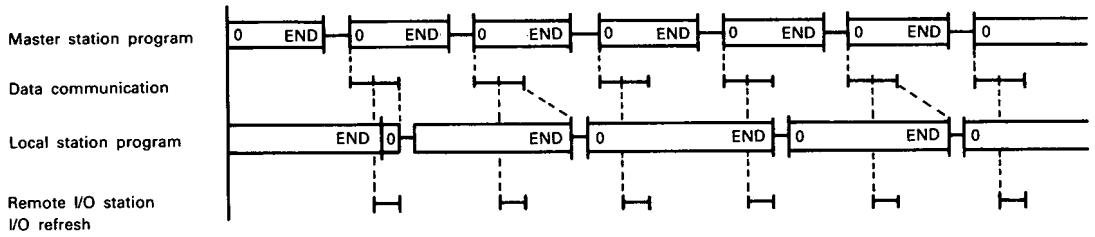
(4) In general data processing, the PC CPU and peripheral or special function module handshake to perform the following:

1) Batch write (Buffer memory, sequence program, parameter data)
2) Batch read (Buffer memory, sequence program, parameter data)
3) Monitoring
4) Testing
5) Remote run/stop/pause

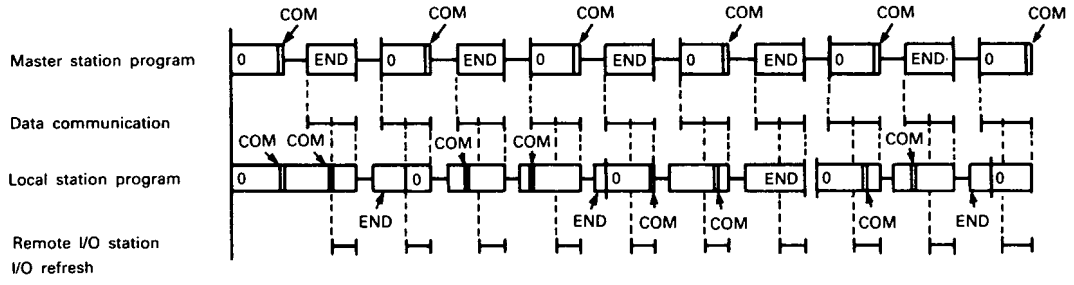
Execution Conditions

(1) Data communication using the COM instruction

1) Example without using the COM instruction



2) Example using the COM instruction



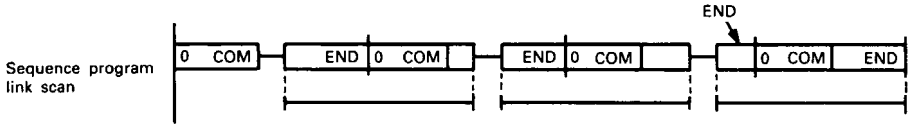
3) By using the COM instruction in the master station, data communication can be made faster as the number of data communication times with the remote I/O station can be increased unconditionally as shown in Example 2).

4) Data may not be received as shown in Example 1) when the scan time of the local station sequence program is longer than that of the master station sequence program. By using the COM instruction in the local station, data can be received securely.

5) By using the COM instruction in the local station, a link refresh is made every time the local station receives the master station command between:

(a) Step 0 and COM instruction
(b) COM instruction and COM instruction
(c) COM instruction and END instruction

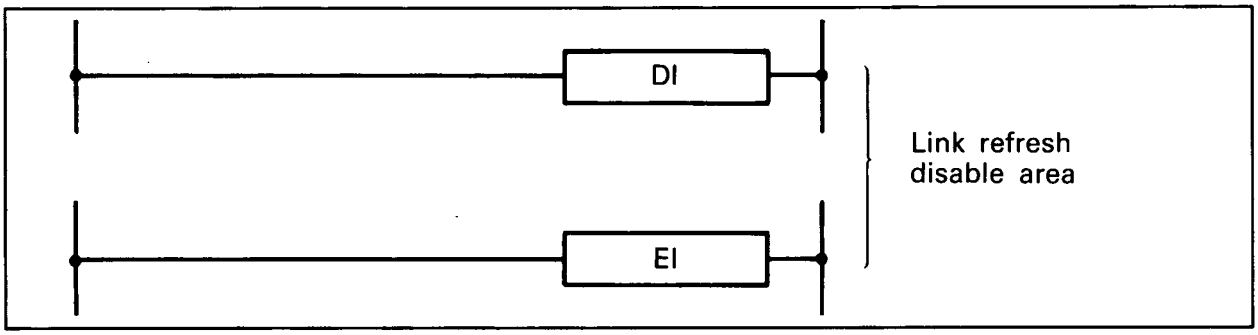
(2) Even if the COM instruction is used in the master station, data communication cannot be made faster when the link scan time is longer than the master station sequence program scan time.



5.9.2 Link refresh enable, disable (EI, DI) (Valid with M9053 on)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	~~A3H~~

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

DI

(1) Disables link refresh until the EI instruction is executed. Valid when M9053 is on.

(2) Sequence processing is started with link refresh enabled.

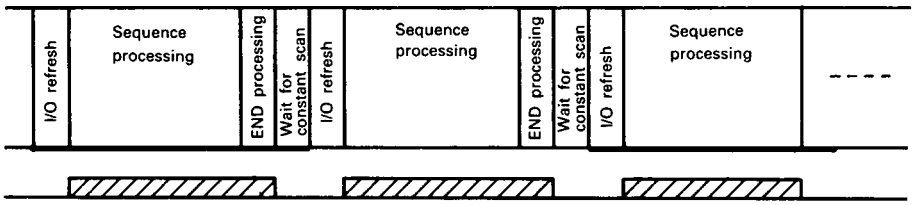
(3) Link refresh is always enabled during END processing.

EI

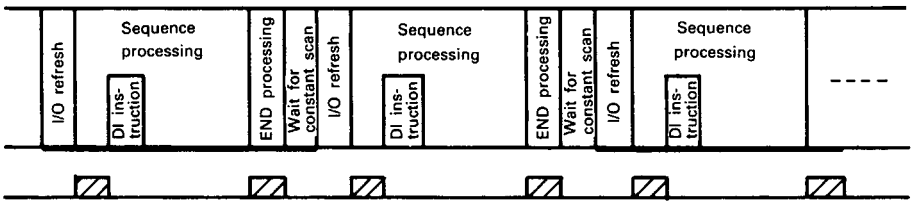
(1) Enables link refresh. Valid when M9053 is on.

Execution Conditions

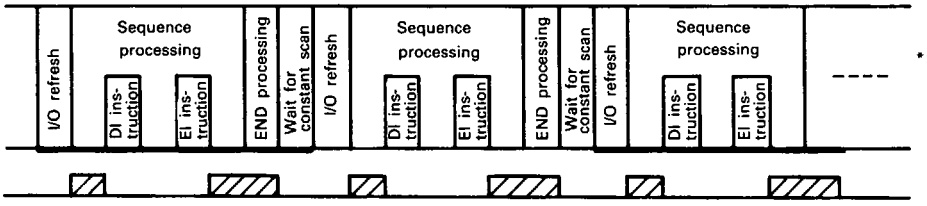
(1) EI/DI instructions are not used



(2) EI instruction is used



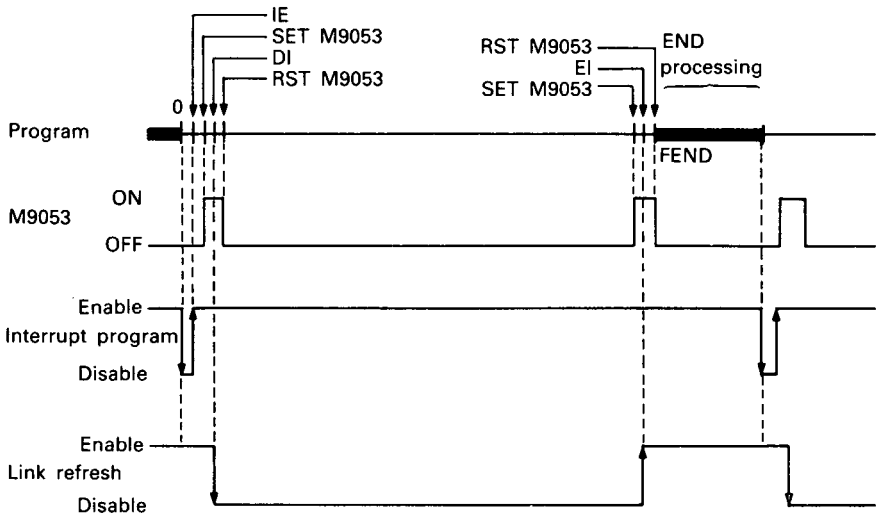
(3) EI/DI instructions are used

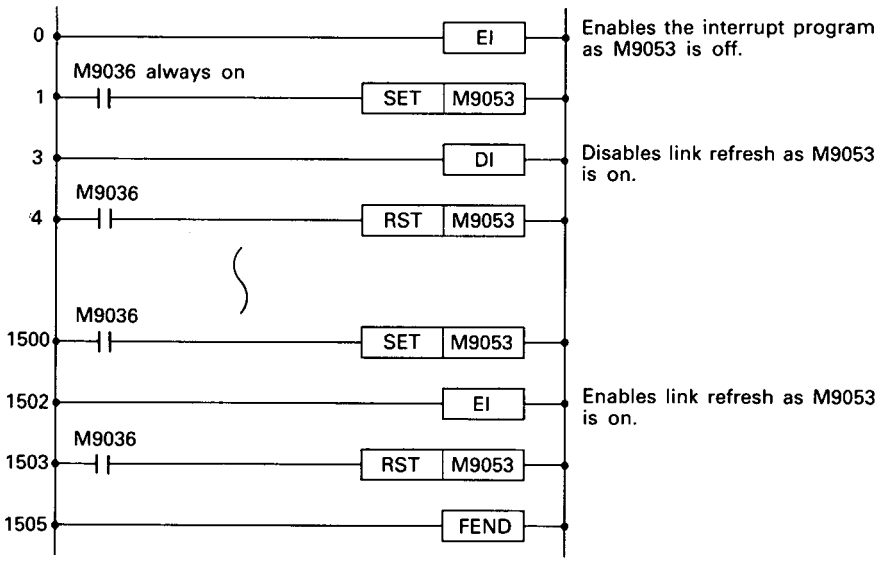


*: • ▨ indicates that link processing is possible.
• There is no wait period for constant scan when the constant scan facility is not specified.
• There is no I/O refresh time in direct mode.

Program Example

The following program allows the interrupt program to be called at any time and link refresh to be disabled until the EI instruction is executed before the FEND instruction is executed.





IMPORTANT

1) Processing is started with link refresh enabled.

2) The interrupt program is started with interrupt disabled.

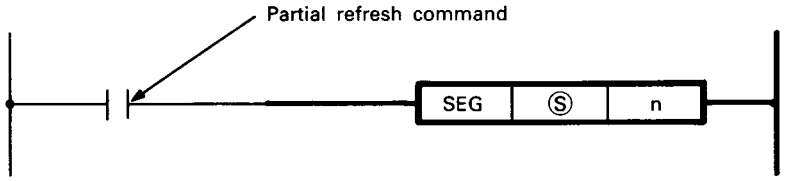
3) When the EI/DI instruction is executed, the operating system judges whether EI/DI processing is performed for the link refresh or interrupt program by the on/off status of M9053.

4) After the EI/DI instruction is executed, M9053 may either be on or off.

5.9.3 Partial refresh (SEG) (Valid with M9052 on)

Processing Unit	Applicable CPU			
	A1N	A2N	A3N	A3H
———				

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n						○																K1 to K4	7					
Ⓢ	○	○																										



Setting data

Ⓢ	Head device number for refresh
n	Number of points refreshed. Set in blocks of 8 points.

Functions

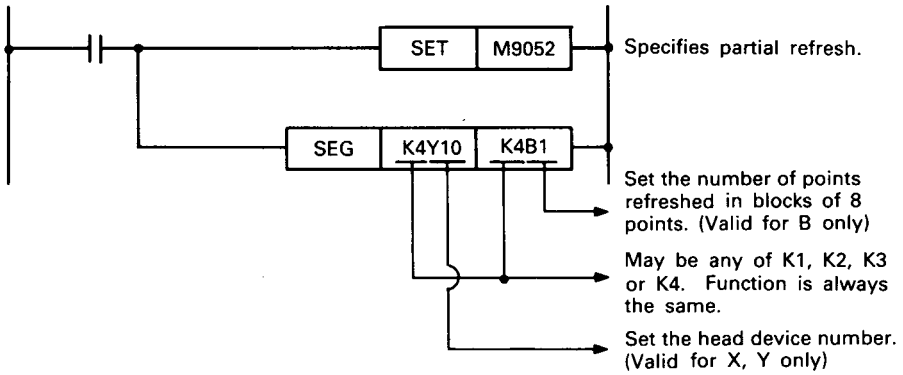
(1) Refreshes the specified devices part way through a scan.

(2) Used to change the on/off status of input (X), output (Y) during a scan in refresh mode.

(3) Allows pulse signals to be output during a scan.

Execution Conditions

(1) Data must be set as shown below:



(2) Setting the head device number May be X/Yn0 or X/Yn8.

IB (NA) 66147-A

(3) Setting the number of points refreshed

The actual points refreshed are (set value) × 8 points and may be up to 2048 points maximum.

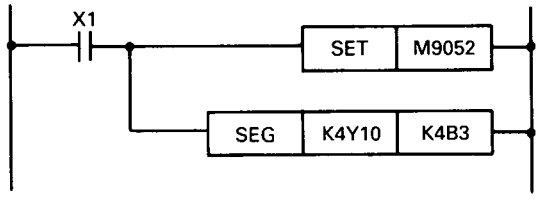
B1 = 8 points
B2 = 16 points
⋮ ⋮
BA = 80 points
BB = 80 points
⋮ ⋮
B10 = 128 points
⋮ ⋮
BFF = 2048 points

(4) Partial refresh processing is still performed if the SEG instruction is executed with the CPU set in X/Y direct mode, but in this case, input (X)/output (Y) ON/OFF status does not change.

(5) Setting B0 (0 point) refreshes all devices in the unit, beginning with the head device number specified.

Program Examples

1) The following example refreshes Y10 to Y27.

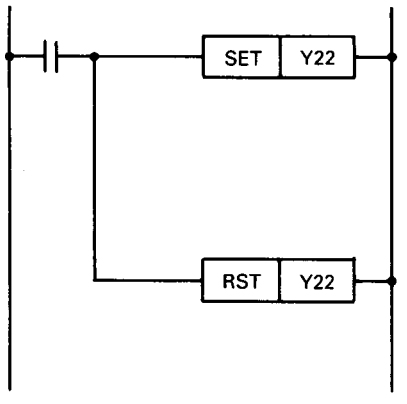


Coding

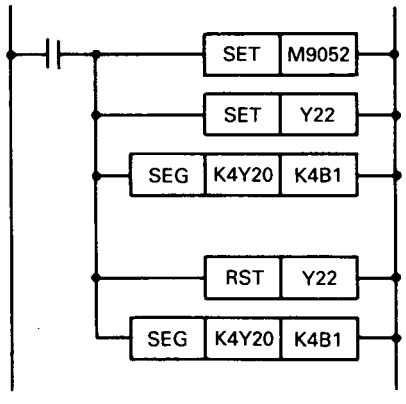
Step Number	Instruction	Device			
0	LD	X1			
1	SET	M9052			
2	SEG	K4Y10	K4B3		
9	END				

2) Pulse output using the SET and RST instructions in direct mode should be changed as shown below when the I/O control is changed to refresh mode.

Direct mode



Refresh mode



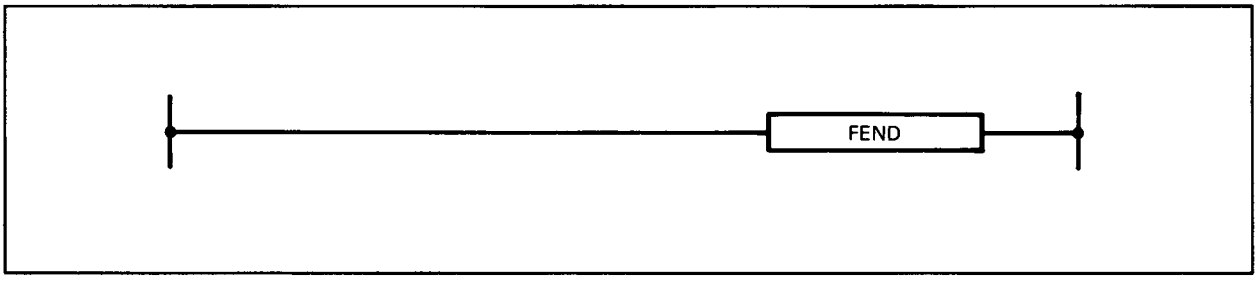
5

5.10 Termination Instructions

5.10.1 Main routine program termination (FEND)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1				○	○

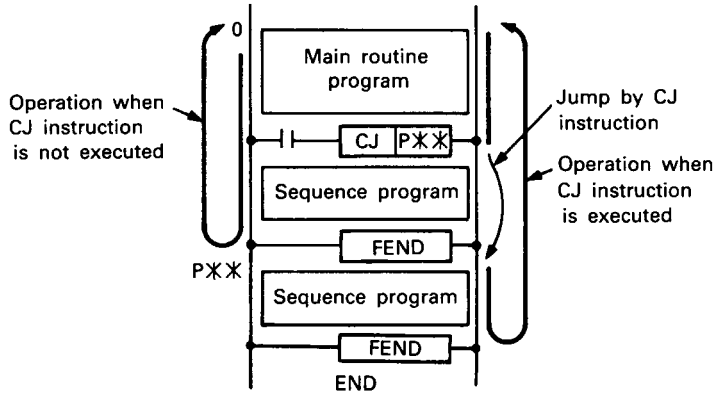


Functions

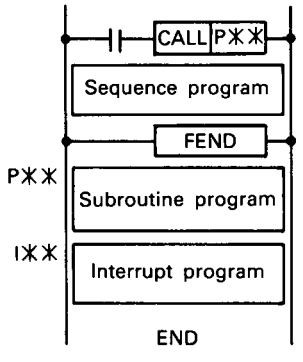
(1) Terminates the main routine program.

(2) When the FEND instruction is executed, the PC returns to step 0 after the processing (such as timer/counter processing and self-diagnostic check) after the execution of END instruction, and resumes operation from step 0.

(3) The sequence program located after FEND instruction can also be displayed on the A6GPP, A6PHP, A6HGP. (The A6GPP, A6PHP, A6HGP displays a circuit up to the END instruction.)



(a) By use of CJ instruction



(b) There are subroutine program and interrupt program

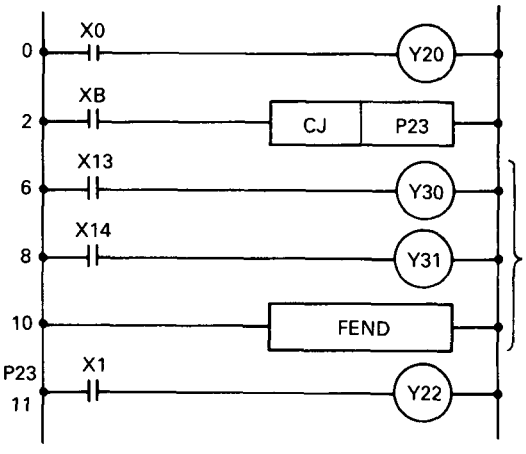
Grammatical Errors

In the following cases, grammatical error occurs and the PC stops its operation.

○After the CALL(P) instruction is executed, the FEND instruction has been executed before executing the RET instruction.
○After the FOR instruction is executed, the FEND instruction has been executed before executing the NEXT instruction.

Program Example

1) Program which uses the CJ instruction.



When XB is on, jump is made to label P23 and execution is performed from the next step to P23.

Executed when XB is off.

Indicates the end of sequence program when XB is off.

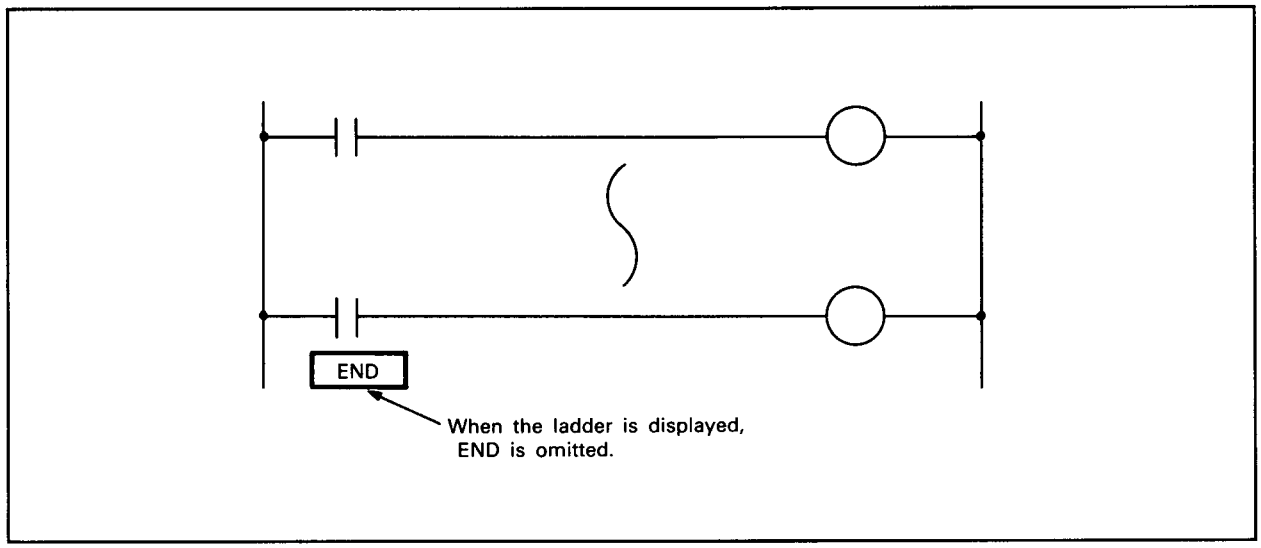
Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	Y20			
2	LD	XB			
3	CJ	P23			
6	LD	X13			
7	OUT	Y30			
8	LD	X14			
9	OUT	Y31			
10	FEND				
11	P23				
12	LD	X1			
13	OUT	Y22			
14	END				

5.10.2 Sequence program termination (END)

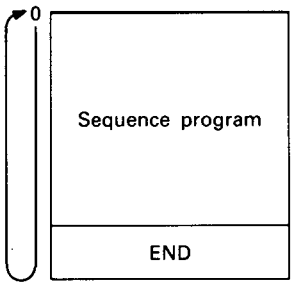
Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



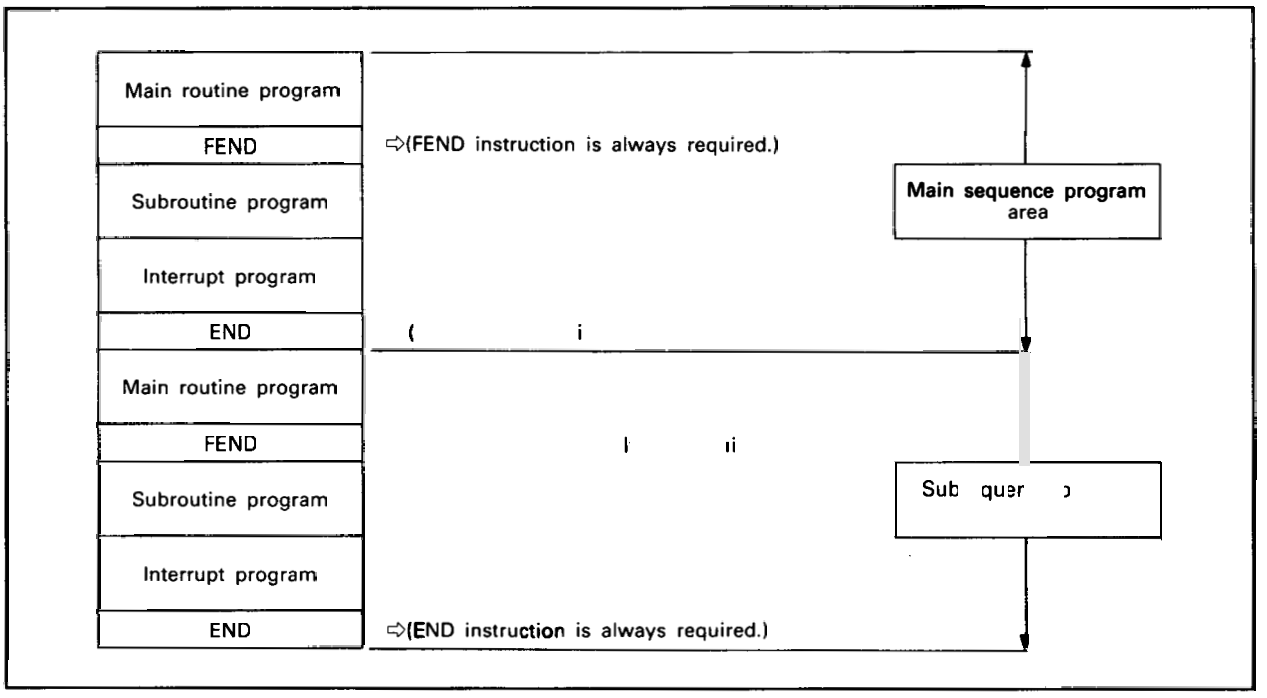
Functions

(1) This instruction indicates the end of program. At this step, the scan returns to step 0.



(2) The END instruction cannot be used midway through the sequence program or subsequence program. If END processing is necessary halfway through the program, use the FEND instruction.

(3) Use the END and FEND instructions in the sequence program, subroutine program, interrupt program, and subsequence program as shown below.



Grammatical Errors

In the following cases, grammatical error occurs and the PC stops its operation.

1) Jump has been made to a step below the END instruction by the CJ, SCJ, or JMP instruction.

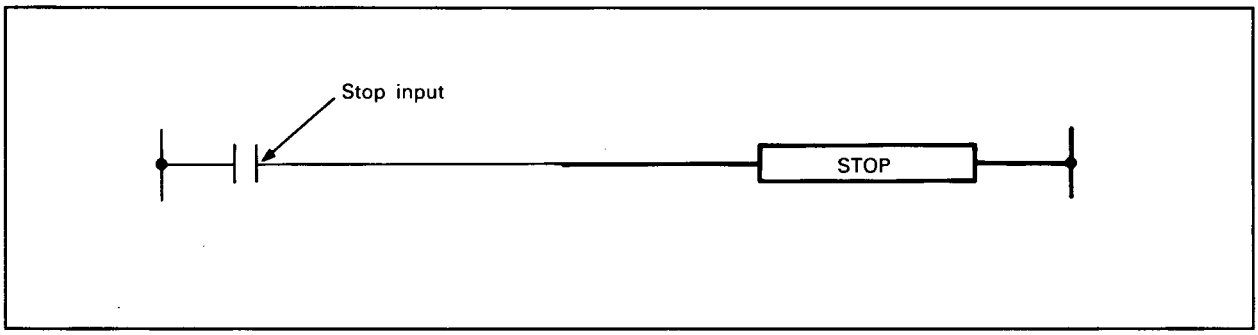
2) The subroutine program or interrupt program located below the END instruction has been executed.

5.11 Other Instructions

5.11.1 Sequence program stop (STOP)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

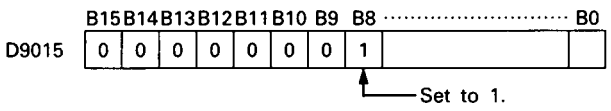
Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

(1) When the stop input turns on, resets the outputs Y and stops the operation of PC. (The same function as when the RUN key switch is moved to the STOP position)

(2) When the STOP instruction is executed, B8 of the special register D9015 is set to 1.



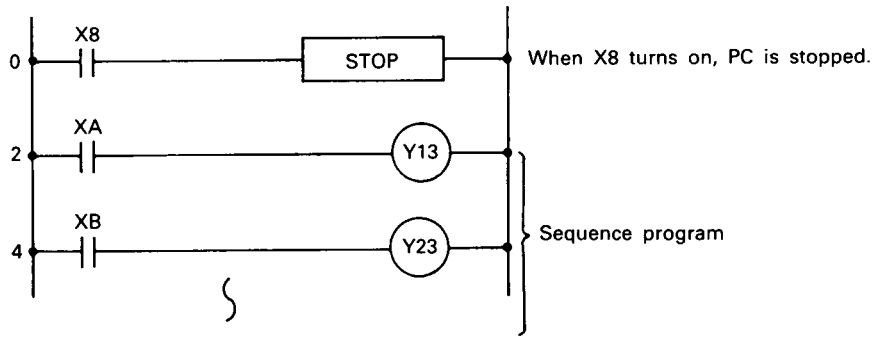
(3) To resume the operation of PC after the execution of STOP instruction, move the RUN key switch from the RUN to the STOP position and then move it to the RUN position again.

(4) Even if the RESET switch is moved to the "LATCH CLEAR" position when the STOP instruction has been executed, latch clear is not executed. To execute the latch clear, move the RUN key switch to the STOP position and then move the RESET switch to the "LATCH CLEAR" position.

(5) Do not provide the STOP instruction in the interrupt program, subroutine program, and FOR/NEXT.

Program Examples

1) Program which stops the PC when X8 turns on.



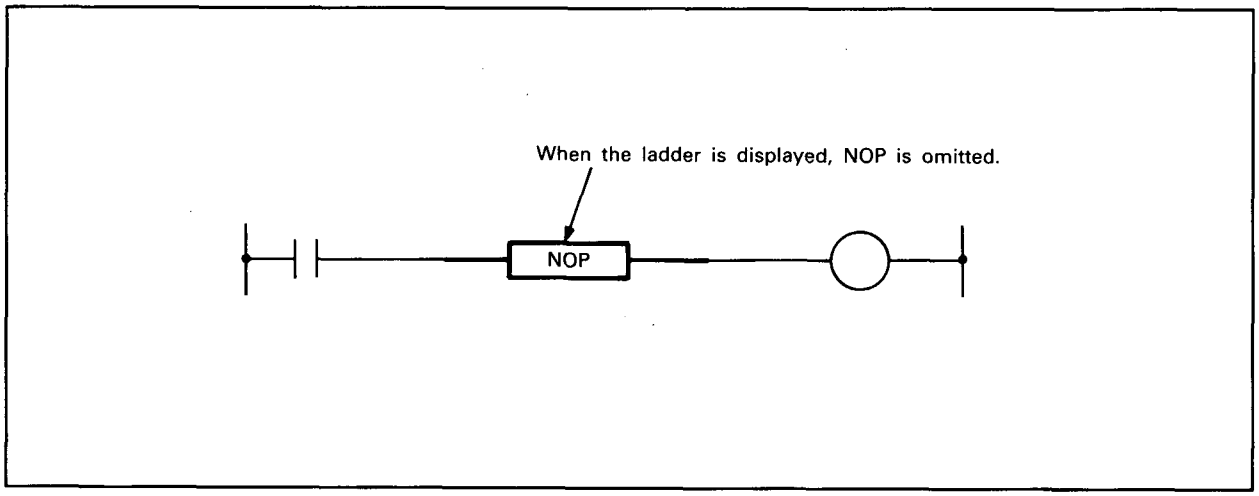
Coding

Step Number	Instruction	Device			
0	LD	X8			
1	STOP				
2	LD	XA			
3	OUT	Y13			
4	LD	XB			
5	OUT	Y23			

5.11.2 No operation (NOP)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



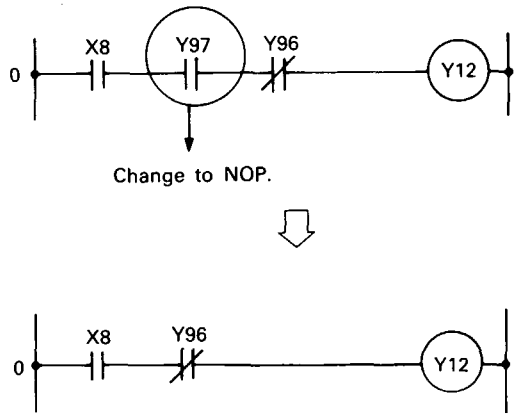
Functions

(1) This is a no operation instruction and has no effect on the previous operation.

(2) NOP is used in the following cases:
1) To provide space for debugging of sequence programs.
2) To delete an instruction without changing the number of steps. (Overwrite with NOP)
3) To delete an instruction temporarily.

Program Examples

1) Short of contact (AND, ANI)



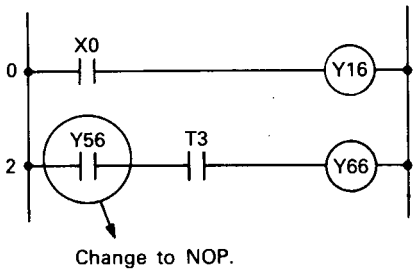
Coding

Step Number	Instruction	Device			
0	LD	X8			
1	AND	Y97			
2	ANI	Y96			
3	OUT	Y12			
4	END				

Coding

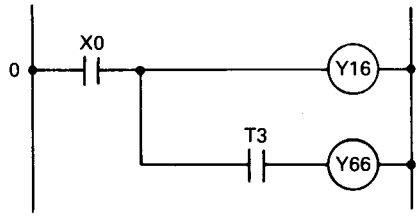
Step Number	Instruction	Device			
0	LD	X8			
1	NOP				
2	ANI	Y96			
3	OUT	Y12			
4	END				

2) Short of contact (LD, LDI): If LD or LDI is changed to NOP, the circuit changes completely. Therefore, caution should be exercised.



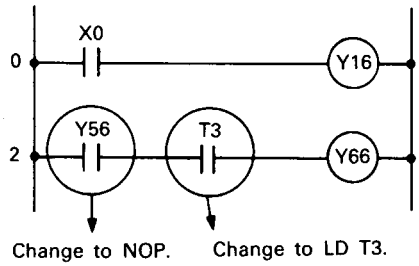
Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	Y16			
2	LD	Y56			
3	AND	T3			
4	OUT	Y66			
5	END				



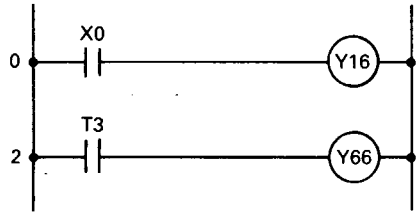
Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	Y16			
2	NOP				
3	AND	T3			
4	OUT	Y66			
5	END				



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	Y16			
2	LD	Y56			
3	AND	T3			
4	OUT	Y66			
5	END				



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	OUT	Y16			
2	NOP				
3	LD	T3			
4	OUT	Y66			
5	END				

MEMO

6. BASIC INSTRUCTIONS

The basic instructions are instructions which are capable of handling numeric data expressed in 16 bits and 32 bits, and are classified into the following instructions.

Classification of Basic Instructions	Description	Ref. Page
Comparison operation instruction	Comparison such as =, >, and <	6-2 to 6-7
Arithmetic operation instruction	Addition, subtraction, multiplication, and division in BIN and BCD.	6-8 to 6-37
BCD ⟷ BIN conversion instruction	Conversion from BCD to BIN and from BIN to BCD	6-38 to 6-44
Data transfer instruction	Transfer of specified data	6-45 to 6-55

6.1 Comparison Operation Instructions

(1) The comparison operation instructions make numerical magnitude comparisons (such as =, >, and <) between two pieces of data. They are handled as a contact, and turn on when their preceding condition holds.

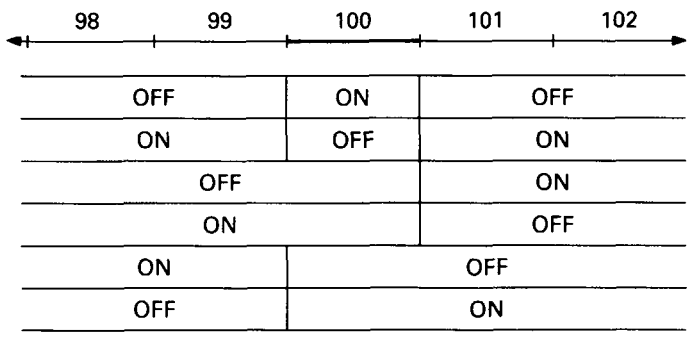
(2) The application of comparison operation instruction is the same as that of the contact instruction for the corresponding sequence instruction as indicated below:

- LD, LDI: LD =, LDD =
- AND, ANI: AND =, ANDD =
- OR, ORI: OR =, ORI =

(3) The comparison operation instructions are available in the following 36 types:

Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page
=	LD=	6-4 to 6-5	>	LD>	6-4 to 6-5	<	LD<	6-4 to 6-5
	AND=			AND>			AND<	
	OR=			OR>			OR<	
	LDD=	6-6 to 6-7		LDD>	6-6 to 6-7		LDD<	6-6 to 6-7
	ANDD=			ANDD>			ANDD<	
	ORD=			ORD>			ORD<	
≠	LD<>	6-4 to 6-5	≦	LD<=	6-4 to 6-5	≧	LD>=	6-4 to 6-5
	AND<>			AND<=			AND>=	
	OR<>			OR<=			OR>=	
	LDD<>	6-6 to 6-7		LDD<=	6-6 to 6-7		LDD>=	6-6 to 6-7
	ANDD<>			ANDD<=			ANDD>=	
	ORD<>			ORD<=			ORD>=	

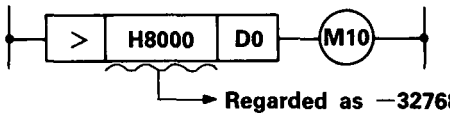
(4) The conditions, by which the comparison operation instructions turn on, are as shown below.



POINT

(1) The comparison instructions make the comparison, regarding the specified data as a BIN value. For this reason, in the case of comparison made in BCD value or hexadecimal, when a numeric value (8 to F) having 1 at the highest bit (B15 in a 16-bit instruction or B31 in a 32-bit instruction) is specified, the comparison is made with the numeric value regarded as the negative of the BIN value.

Example

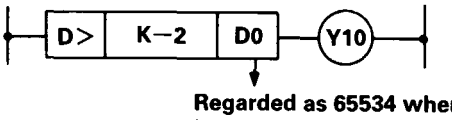


H8000 D0 M10

Regarded as −32768 in BIN value.

(2) When the comparison of 32-bit data is made, specify the numeric value using the 32-bit instruction.

Example



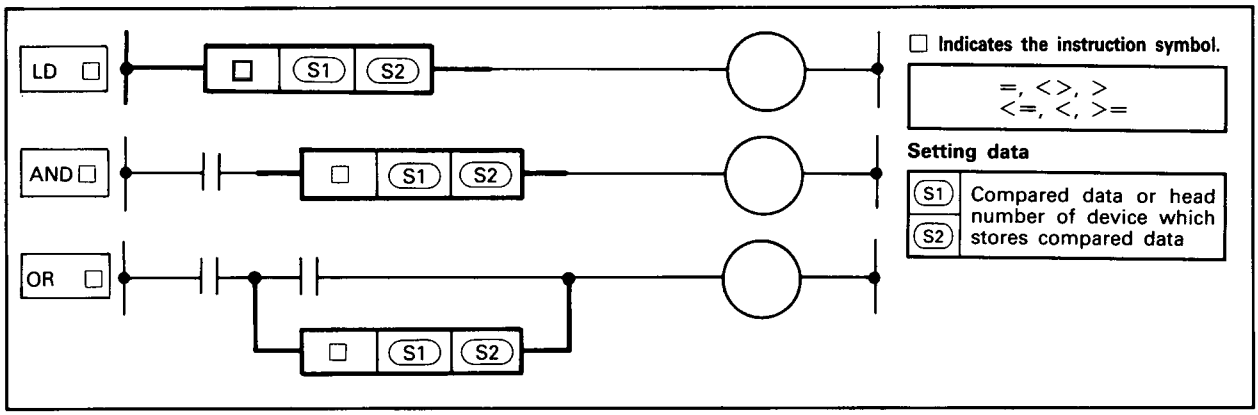
K−2 D0 Y10

Regarded as 65534 when (−1) is stored into D0 by 16-bit instruction.

6.1.1 16-bit data comparison (=, < >, >, <=, <, >=)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5/7	○	○		○	○
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										



Functions

(1) Handled as a "N/O" contact and used for the comparison of 16 bits.

(2) The comparison operation result is as shown below:

Instruction Symbol in □	Condition	Comparison Operation Result	Instruction Symbol in □	Condition	Comparison Operation Result
=	(S1) = (S2)	Continuity status	=	(S1) ≠ (S2)	Non-Continuity status
<>	(S1) ≠ (S2)		<>	(S1) = (S2)	
>	(S1) > (S2)		>	(S1) ≦ (S2)	
<=	(S1) ≦ (S2)		<=	(S1) > (S2)	
<	(S1) < (S2)		<	(S1) ≧ (S2)	
>=	(S1) ≧ (S2)		>=	(S1) < (S2)	

Execution Conditions

The execution conditions of LD□, AND□, and OR□ are as indicated below.

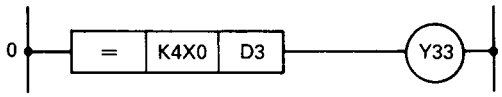
Instruction	Execution Condition
LD□	Executed per scan.
AND□	Executed only when the preceding contact instruction is on.
OR□	Executed per scan.

REMARKS

The number of steps is seven in the following cases:
- Index qualification has been performed.
- The digit specification of bit device is not K4.
- The head number of bit device is not a multiple of 8.
  A multiple of 16 when the A3HCPU is used.

Program Examples

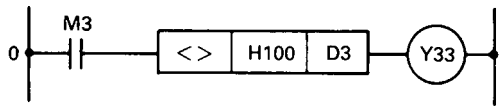
1) Program which compares the data of X0 to F and the data f D3.



Coding

Step Number	Instruction	Device			
0	LD=	K4X0	D3		
5	OUT	Y33			
6	END				

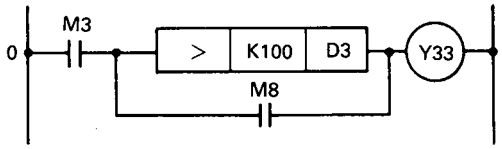
2) Program which compares the BCD value 100 and the data of D3.



Coding

Step Number	Instruction	Device			
0	LD	M3			
1	AND< >	H100	D3		
6	OUT	Y33			
7	END				

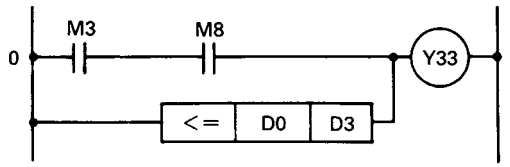
3) Program which compares the BIN value 100 and the data of D3.



Coding

Step Number	Instruction	Device			
0	LD	M3			
1	LD>	K100	D3		
6	OR	M8			
7	ANB				
8	OUT	Y33			
9	END				

4) Program which compares the data of D0 and that of D3.



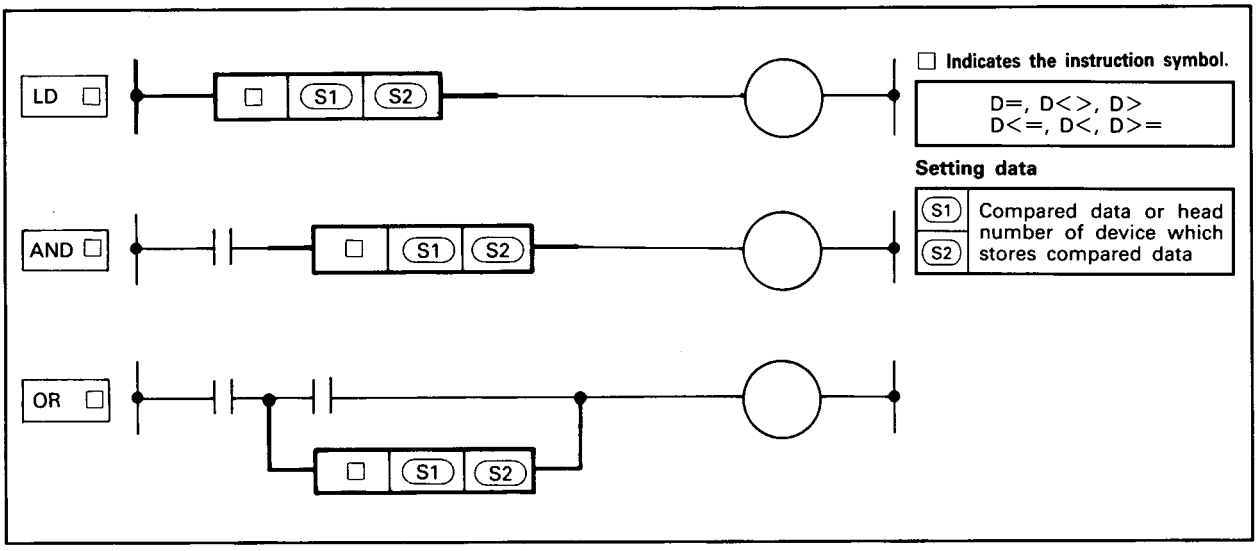
Coding

Step Number	Instruction	Device			
0	LD	M3			
1	AND	M8			
2	OR<=	D0	D3		
7	OUT	Y33			
8	END				

6.1.2 32-bit data comparison (D=, D< >, D>, D<=, D<, D>=)

Processing Unit	Applicable CPU			
32 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1 to K8	11		○		○	○
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○										



Functions

(1) Handled as a "N/O" contact and used for the comparison of 32 bits.

(2) The comparison operation result is as shown below:

Instruction Symbol in □	Condition	Comparison Operation Result	Instruction Symbol in □	Condition	Comparison Operation Result
D=	(S1) = (S2)	Continuity status	D=	(S1) ≠ (S2)	Non-Continuity status
D<>	(S1) ≠ (S2)		D<>	(S1) = (S2)	
D>	(S1) > (S2)		D>	(S1) ≦ (S2)	
D<=	(S1) ≦ (S2)		D<=	(S1) > (S2)	
D<	(S1) < (S2)		D<	(S1) ≧ (S2)	
D>=	(S1) ≧ (S2)		D>=	(S1) < (S2)	

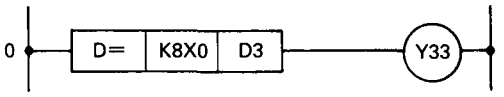
Execution Conditions

The execution conditions of LD□, AND□, and OR□ are as indicated below.

Instruction	Execution Condition
LD□	Executed per scan.
AND□	Executed only when the preceding contact instruction is on.
OR□	Executed per scan.

Program Examples

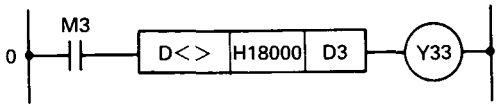
1) Program which compares the data of X0 to 1F and the data of D3 and D4.



Coding

Step Number	Instruction	Device			
0	LDD=	K8X0	D3		
11	OUT	Y33			
12	END				

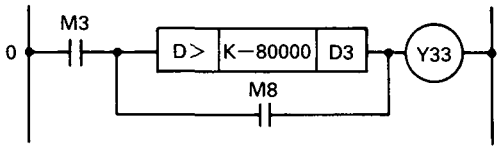
2) Program which compares the BCD value 18000 and the data of D3 and D4



Coding

Step Number	Instruction	Device			
0	LD	M3			
1	ANDD<>	H18000	D3		
12	OUT	Y33			
13	END				

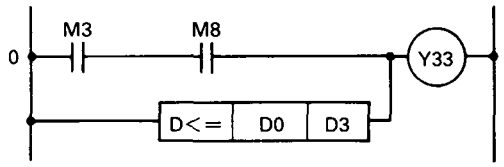
3) Program which compares the BIN value −80000 and the data of D3 and D4.



Coding

Step Number	Instruction	Device			
0	LD	M3			
1	LDD>	K-80000	D3		
12	OR	M8			
13	ANB				
14	OUT	Y33			
15	END				

4) Program which compares the data of D1 and D0 and that of D3 and D4.



Coding

Step Number	Instruction	Device			
0	LD	M3			
1	AND	M8			
2	ORD<=	D0	D3		
13	OUT	Y33			
14	END				

6.2 Arithmetic Operation Instructions

The arithmetic operation instructions are instructions which perform the addition, subtraction, multiplication, and division of two BIN data or BCD data. The arithmetic operation instructions are available in the following 56 types.

Classification	BIN		BCD	
	Instruction Symbol	Ref. Page	Instruction Symbol	Ref. Page
+	+	6-10 to 6-12	B+	6-22 to 6-24
	+P	6-10 to 6-12	B+P	6-22 to 6-24
	D+	6-13 to 6-15	DB+	6-25 to 6-27
	D+P	6-13 to 6-15	DB+P	6-25 to 6-27
−	−	6-10 to 6-12	B−	6-22 to 6-24
	−P	6-10 to 6-12	B−P	6-22 to 6-24
	D−	6-13 to 6-15	DB−	6-25 to 6-27
	D−P	6-13 to 6-15	DB−P	6-25 to 6-27
*	*	6-16 to 6-18	B*	6-28 to 6-30
	*P	6-16 to 6-18	B*P	6-28 to 6-30
	D*	6-19 to 6-21	DB*	6-31 to 6-33
	D*P	6-19 to 6-21	DB*P	6-31 to 6-33
/	/	6-16 to 6-18	B/	6-28 to 6-30
	/P	6-16 to 6-18	B/P	6-28 to 6-30
	D/	6-19 to 6-21	DB/	6-31 to 6-33
	D/P	6-19 to 6-21	DB/P	6-31 to 6-33
+1	INC	6-34 to 6-35		
	INCP	6-34 to 6-35		
	DINC	6-36 to 6-37		
	DINCP	6-36 to 6-37		
−1	DEC	6-34 to 6-35		
	DECP	6-34 to 6-35		
	DDEC	6-36 to 6-37		
	DDECP	6-36 to 6-37		

Arithmetic operation with BIN (Binary)

- If the operation result of an addition instruction exceeds 32767 (2147483647 in the case of a 32-bit instruction), the result becomes a negative value.

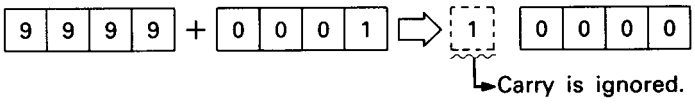
- If the operation result of a subtraction instruction is less than −32768 (−2147483648 in the case of a 32-bit instruction), the result becomes a positive value.

- The operation of a positive value and a negative value is as follows:

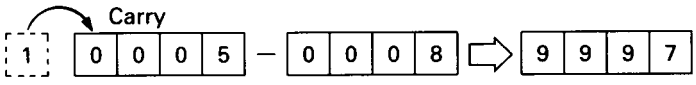
```
  5 + 8     → 13
  5 − 8     → −3
  5 × 3     → 15
 −5 × 3     → −15
 −5 × (−3)  → 15
 −5 ÷ 3     → −1 and remainder −2
  5 ÷ (−3)  → −1 and remainder 2
 −5 ÷ (−3)  → 1 and remainder −2
```

Arithmetic operation with BCD

- If the operation result of an addition instruction has exceeded 9999 (99999999 in the case of a 32-bit instruction), carry is ignored.



- When the subtrahend is less than the minuend in the subtraction instruction, the following occurs.

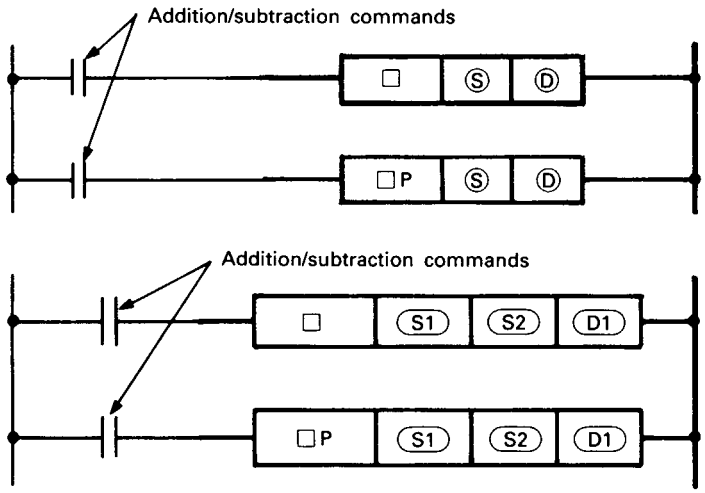


6

6.2.1 BIN 16-bit addition, subtraction (+, +P, −, −P)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1	5					
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				to		○	○		○	○
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K4	7					
(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												



□ Indicates the instruction symbol.

+, −

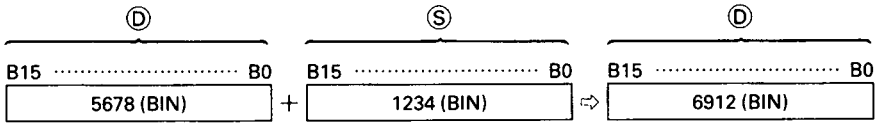
Setting data

Ⓢ	Addend/subtrahend or head device number storing addend/subtrahend
Ⓓ	Head device number storing augend/minuend
(S1)	Augend/minuend or head device number storing augend/minuend
(S2)	Addend/subtrahend or head device number storing addend/subtrahend
(D1)	Head device number which will store the operation result

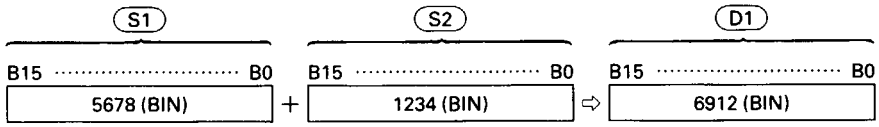
Functions

+

(1) Performs the addition of BIN data specified at Ⓓ and the BIN data specified at Ⓢ, and stores the addition result into the device specified at Ⓓ.



(2) Performs the addition of BIN data specified at (S1) and the BIN data specified at (S2), and stores the addition result into the device specified at (D1).



(3) At Ⓢ, (S1), (S2) and Ⓓ, −32768 to 32767 (BIN 16 bits) can be specified.

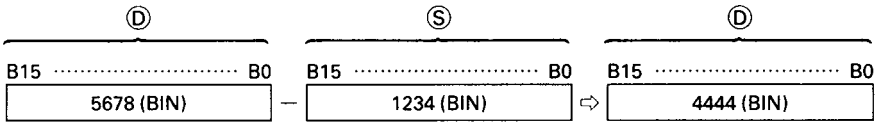
(4) The judgement of whether the datas of Ⓢ, (S1), (S2) and Ⓓ are positive or negative is made at the highest bit (B15).

0 ····· Positive
1 ····· Negative

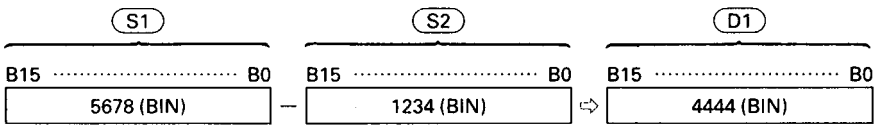
(5) When the 0the bit has underflown, the carry flag does not turn on.
When the 15the bit has overflown, the carry flag does not turn on.

[−]

(1) Performs the subtraction of BIN data specified at Ⓓ and the BIN data specified at Ⓢ, and stores the subtraction result into the device specified at Ⓓ.



(2) Performs the subtraction of BIN data specified at (S1) and the BIN data specified at (S2), and stores the subtraction result into the device specified at (D1).

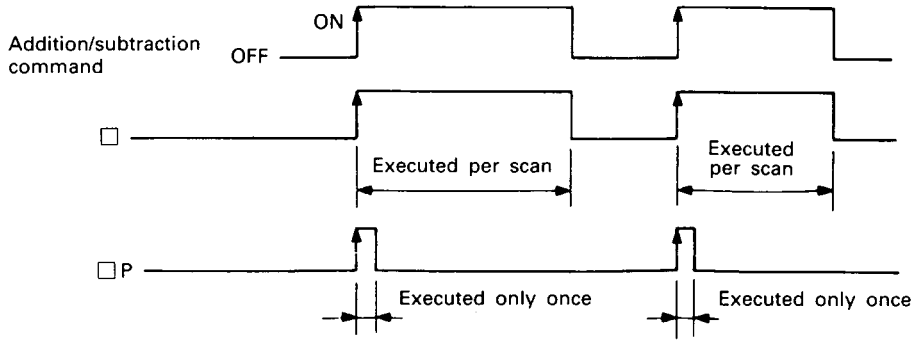


(3) At Ⓢ, (S1), (S2) and Ⓓ, −32768 to 32767 (BIN 16 bits) can be specified.

(4) The judgement of whether the dates of Ⓢ , (S1), (S2) and Ⓓ are positive or negative is made at the highest bit (B15).
0 ····· Positive
1 ····· Negative

(5) When the 0th bit has underflown, the carry flag does not turn on.
When the 15th bit has overflown, the carry flag does not turn on.

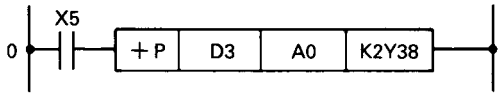
Execution Conditions



Program Examples

[+]

Program which adds the content of A0 to the content of D3 and outputs the result to Y38 to 3F when X5 turns on.

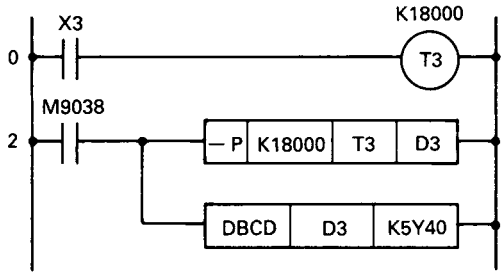


Coding

Step Number	Instruction	Device			
0	LD	X5			
1	+P	D3	A0	K2Y38	
8	END				

[−]

Program which outputs the difference between the set value and present value of timer T3 to Y40 to 4F in BCD.



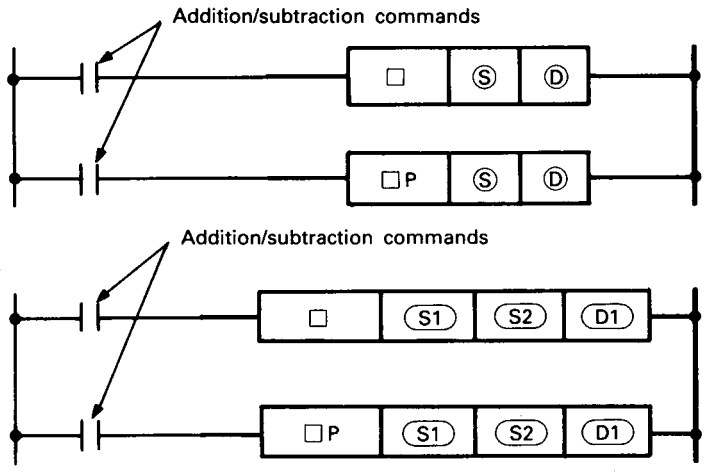
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	OUT	T3	K18000		
2	LD	M9036			
3	−P	K18000	T3	D3	
10	DBCD	D3	K5Y40		
19	END				

6.2.2 BIN 32-bit addition, subtraction (D+, D+P, D−, D−P)

Processing Unit	Applicable CPU			
32 bits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○										
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○							K1	9					
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				to		○	○		○	○
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K8	11					
(D1)		○	○	○	○	○	○	○	○	○	○	○	○		○													



□ Indicates the instruction symbol.

D+, D−

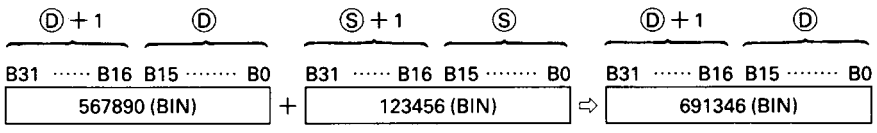
Setting data

Ⓢ	Addend/subtrahend or head device number storing addend/subtrahend
Ⓓ	Head device number storing augend/minuend
(S1)	Augend/minuend or head device number storing augend/minuend
(S2)	Addend/subtrahend or head device number storing addend/subtrahend
(D1)	Head device number which will store the operation result

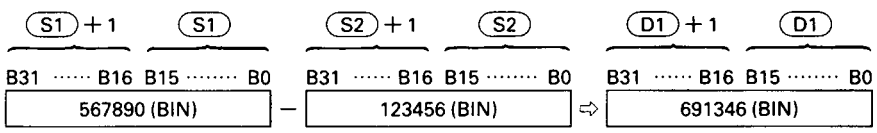
Functions

D+

(1) Performs the addition of BIN data specified at Ⓓ and the BIN data specified at Ⓢ, and stores the addition result into the device specified ad Ⓓ.



(2) Performs the addition of BIN data specified at (S1) and the BIN data specified at (S2), and stores the addition result into the device specified at (D1).



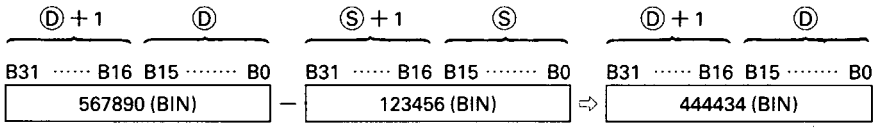
(3) At Ⓢ, (S1), (S2) and Ⓓ, −2147483648 to 2147483647 (BIN 32 bits) can be specified.

(4) The judgement of whether the datas of Ⓢ, (S1), (S2) and Ⓓ are positive or negative is made at the highest bit (B31).
0 ····· Positive
1 ····· Negative

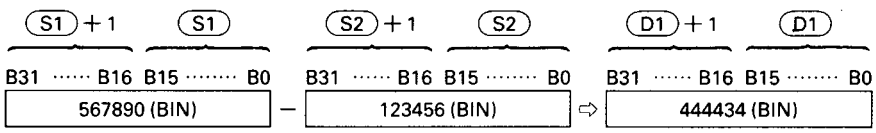
(5) When the 0th bit has underflown, the carry flag does not turn on.
When the 31th bit has overflown, the carry flag does not turn on.

D−

(1) Performs the subtraction of BIN data specified at Ⓓ and the BIN data specified at Ⓢ, and stores the addition result into the device specified at Ⓓ.



(2) Performs the subtraction of device specified at (S1) and the device specified at (S2), and stores the result into the device specified at (D1).

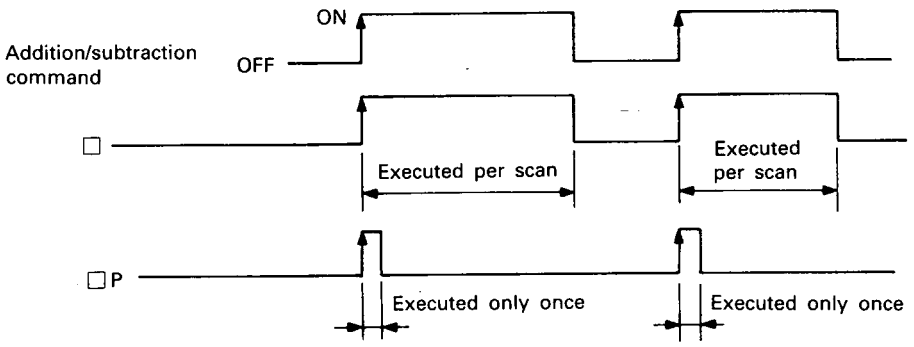


(3) At Ⓢ, (S1), (S2) and Ⓓ, −2147483648 to 2147483647 (BIN 32 bits) can be specified.

(4) The judgement of whether the datas of Ⓢ, (S1), (S2) and Ⓓ are positive or negative is made at the highest bit (B31).
0 ····· Positive
1 ····· Negative

(5) When the 0th bit has underflown, the carry flag does not turn on.
When the 31th bit has overflown, the carry flag does not turn on.

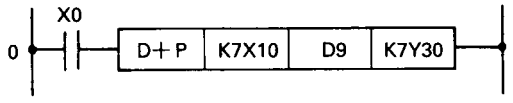
Execution Conditions



Program Examples

D+

Program which adds the 28-bit data of X10 to 2B and the data of D9 and 10, and outputs the result to Y30 to 4B when X0 turns on.

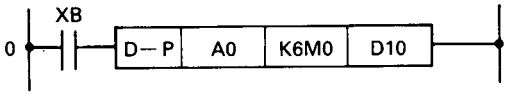


Coding

Step Number	Instruction	Device			
0	LD	X0			
1	D+P	K7X10	D9	K7Y30	
12	END				

D−

The following program subtracts M0 to 23 data from A1 data and stores to D10, D11 when XB is switched on.



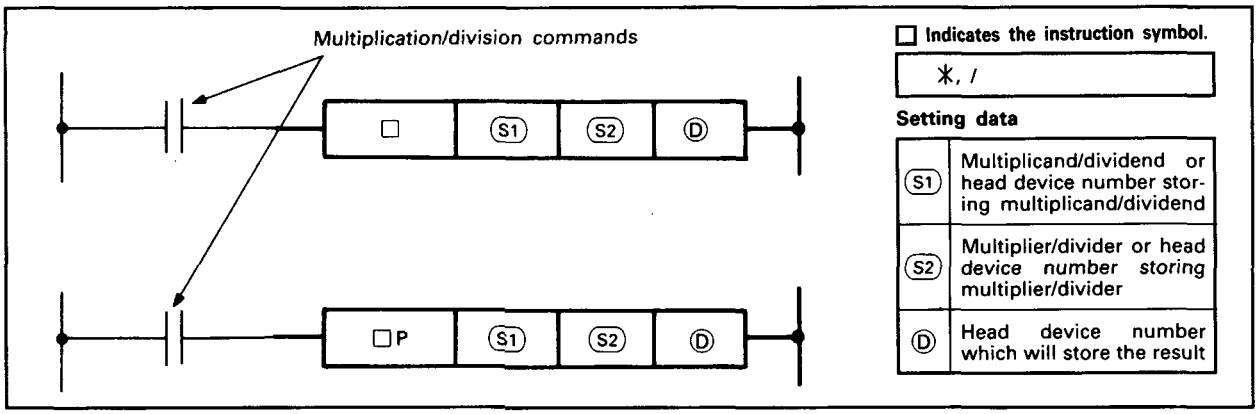
Coding

Step Number	Instruction	Device			
0	LD	XB			
1	D−P	A0	K6M0	D10	
12	END				

6.2.3 BIN 16-bit multiplication, division (✳, ✳P, /, /P)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

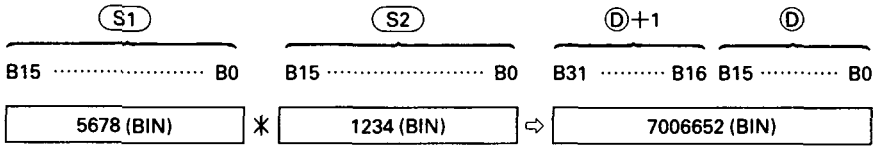
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1						
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				to	7	○	○		○	○
(D)		○	○	○	○	○	○	○	○	○	○	○	○		○							K4						



Functions

✳

(1) Performs the multiplication of BIN data specified at (S1) and the BIN data specified at (S2), and stores the multiplication result into the device specified at (D).



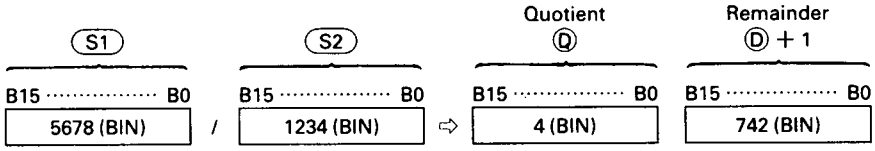
(2) When (D) is a bit device, specify the bits, beginning with the lower bits.
Example
K1: Lower 4 bits (B0 to 3)
K4: Lower 16 bits (B0 to 15)
K8: 32 bits (B0 to 31)

(3) At (S1) and (S2), −32768 to 32767 (BIN 16 bits) can be specified.

(4) The judgement of whether the datas of (S1), (S2), and (D) are positive or negative is made at the highest bit (B15).

/

(1) Performs the division of BIN data specified at (S1) and the BIN data specified at (S2), and stores the result into the device specified at Ⓓ.



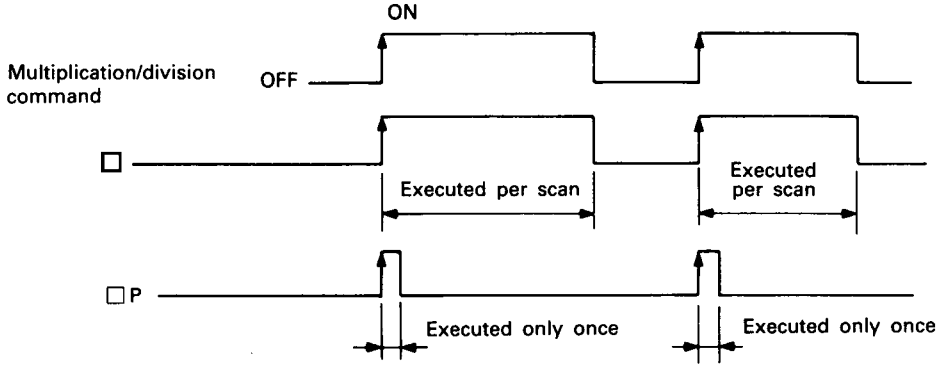
(2) In regards to the operation result, the quotient and remainder are stored by use of 32 bits in the case of word device, and only the quotient is stored by use of 16 bits in the case of bit device.

Quotient: Stored to the lower 16 bits.
Remainder: Stored to the upper 16 bits. (Storable only in the case of word device)

(3) At (S1) and (S2), −32678 to 32767 (BIN 16 bits) can be specified.

(4) The judgement of whether the dates of (S1), (S2), and Ⓓ are positive or negative is made at the highest bit (B15).

Execution Conditions



Operation Errors

In the following case, operation error occurs and the error flag turns on.

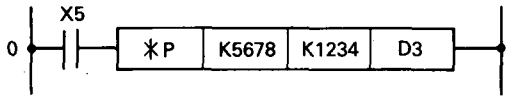
- A1 or V has been specified at Ⓓ.
- The divisor (S2) is 0.

IB (NA) 66147-A

Program Examples

[✳]

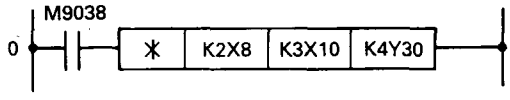
1) Program which stores the multiplication result of 5678 and 1234 in BIN to D3 and 4 when X5 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X5			
1	✳P	K5678	K1234	D3	
8	END				

2) Program which outputs the multiplication result of the BIN data of X8 to F and the BIN data of X10 to 1B to Y30 to 3F.

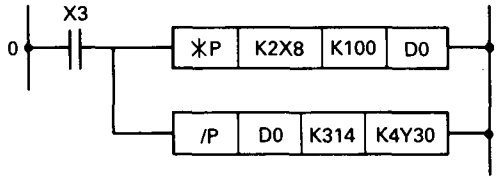


Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	✳	K2X8	K3X10	K4Y30	
8	END				

[/]

Program which outputs the quotient, obtained by dividing the data of X8 to F by 3.14, to Y30 to 3F when X3 turns on.



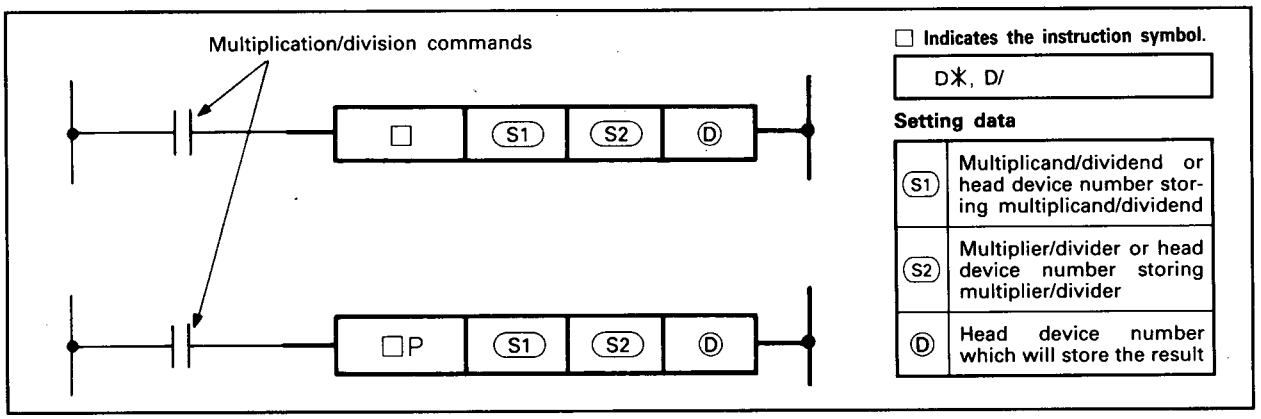
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	✳P	K2X8	K100	D0	
8	/P	D0	K314	K4Y30	
15	END				

6.2.4 BIN 32-bit multiplication, division (D✳, D✳P, D/, D/P)

Processing Unit	Applicable CPU			
32 bits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1						
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				to	11	○	○		○	○
(D)		○	○	○	○	○	○	○	○	○	○	○										K8						



□ Indicates the instruction symbol.

D✳, D/

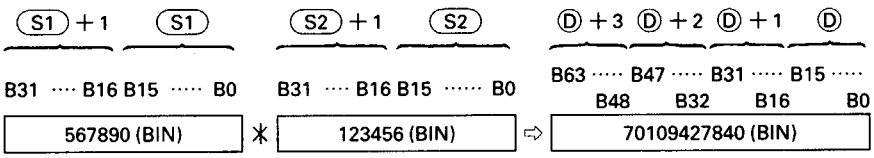
Setting data

(S1)	Multiplicand/dividend or head device number storing multiplicand/dividend
(S2)	Multiplier/divider or head device number storing multiplier/divider
(D)	Head device number which will store the result

Functions

D✳

(1) Performs the multiplication of BIN data specified at (S1) and the BIN data specified at (S2), and stores the multiplication result into the device specified at (D).



(2) When (D) is a bit device, up to the lower 32 bits can be specified and the upper 32 bits cannot be specified.
Example

K1: Lower 4 bits (B0 to 3)
K4: Lower 16 bits (B0 to 15)
K8: 32 bits (B0 to 31)

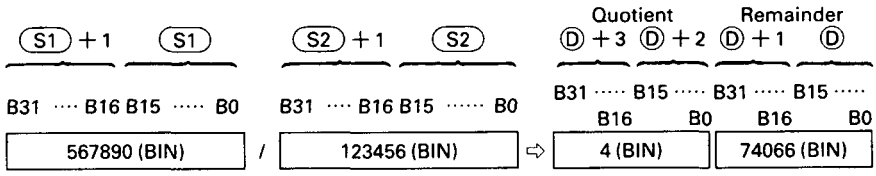
When the upper 32-bit data of multiplication result is required for the bit device, store the data to the word device and then transfer the data ((D) + 2) and ((D) + 3) of word device to the specified bit device.

(3) At (S1) and (S2), −2147483648 to 2147483647 (BIN 32 bits) can be specified.

(4) The judgement of whether the datas of (S1), (S2), and (D) are positive or negative is made at the highest bit (B31).

D/

(1) Performs the division of BIN data specified at (S1) and the BIN data specified at (S2), and stores the division result into the device specified at (D).



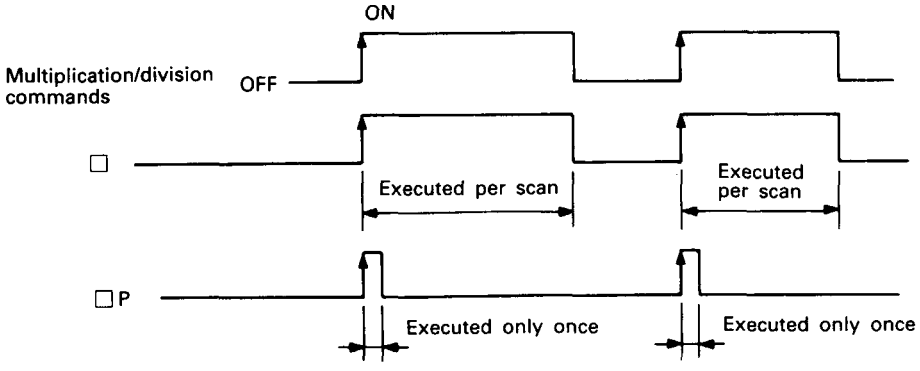
(2) In regards to the operation result, the quotient and remainder are stored by use of 64 bits in the case of word device, and only the quotient is stored by use of lower 32 bits in the case of bit device.

Quotient: Stored to the lower 32 bits.
Remainder: Stored to the upper 32 bits. (Storable only in the case of word device)

(3) At (S1) and (S2), −2147483648 to 2147483647 (BIN 32 bits) can be specified.

(4) The judgement of whether the datas of (S1), (S2), and (D) are positive or negative is made at the highest bit (B31).

Execution Conditions



Operation Errors

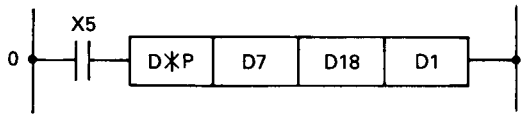
In the following case, operation error occurs and the error flag turns on.

- A1, V are specified in (S1), (S2) and A0, A1, Z, V specified in (D)
- The divisor (S2) is 0.

Program Examples

D✳

Program which stores the multiplication result of the BIN data of D7 and D8 and the BIN data of D18 and D19 to D1 to D4 when X5 turns on.

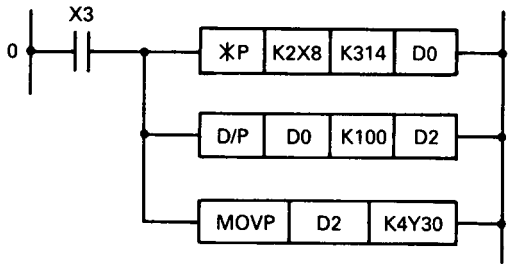


Coding

Step Number	Instruction	Device			
0	LD	X5			
1	D✳P	D7	D18	D1	
12	END				

D/

Program which outputs a value, obtained by multiplying the data of X8 to F by 3.14, to Y30 to 3F when X3 turns on.



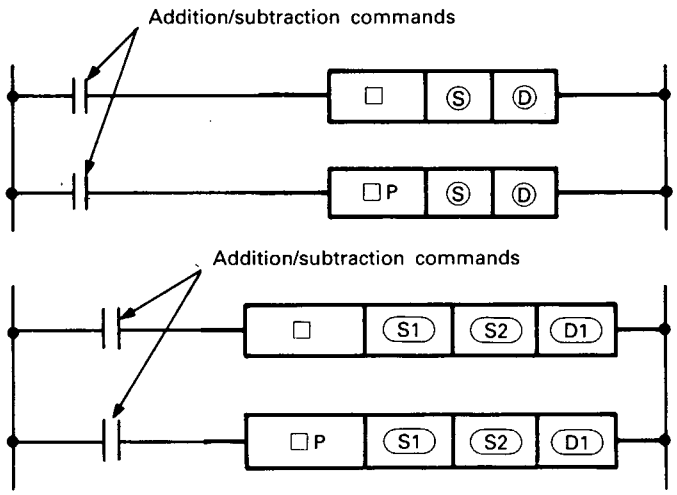
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	✳P	K2X8	K314	D0	
8	D/P	D0	K100	D2	
9	MOVP	D2	K4Y30		
24	END				

6.2.5 BCD 4-digit addition, subtraction (B+, B+P, B−, B−P)

Processing Unit	Applicable CPU			
BCD 4 digits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	7 to 9		○		○	○
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												



□ Indicates the instruction symbol.

B+, B−

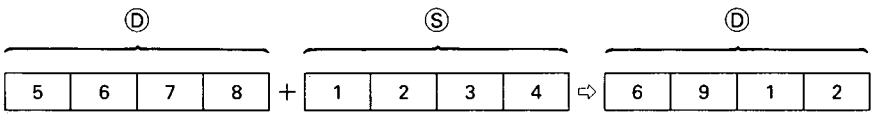
Setting data

Ⓢ	Addend/subtrahend or head device number storing addend/subtrahend
Ⓓ	Head device number storing augend/minuend
(S1)	Augend/minuend or head device number storing augend/minuend
(S2)	Addend/subtrahend or head device number storing addend/subtrahend
(D1)	Head device number which will store the operation result

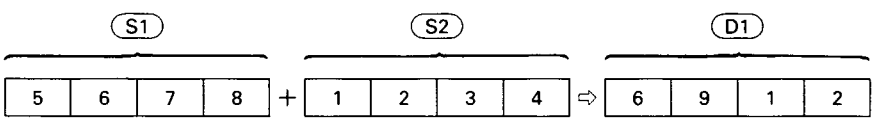
Functions

B+

(1) Performs the addition of BCD data specified at Ⓓ and the BCD data specified at Ⓢ, and stores the addition result into the device specified at Ⓓ.



(2) Performs the addition of BCD data specified at (S1) and the BCD data specified at (S2), and stores the addition result into the device specified at (D1).

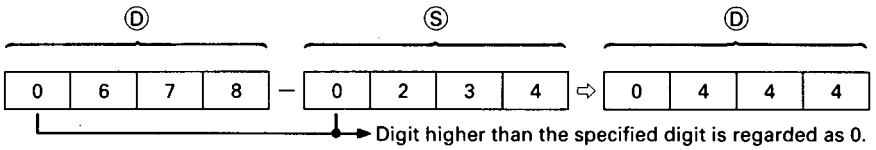


(3) At Ⓢ, (S1), (S2) and Ⓓ, 0 to 9999 (BCD 4 digits) can be specified.

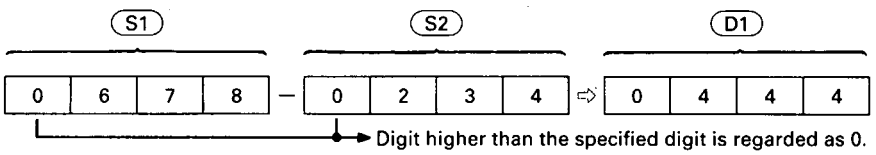
(4) Even if the addition result exceeds 9999, the carry flag does not turn on and the carry digit is ignored.

B—

(1) Performs the subtraction of BCD data specified at Ⓓ and the BCD data specified at Ⓢ, and stores the subtraction result into the device specified at Ⓓ.



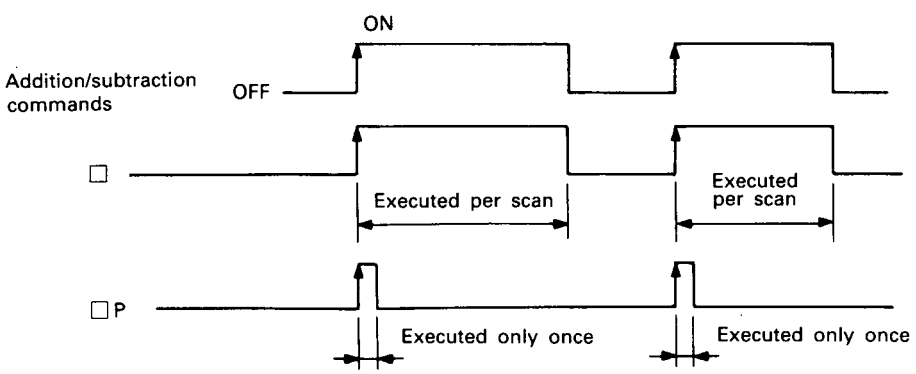
(2) Performs the subtraction of BCD data specified at (S2) and the BCD data specified at (S1), and stores the subtraction result into the device specified at (D1).



(3) At Ⓢ, (S1), (S2) and Ⓓ, 0 to 9999 (BCD 4 digits) can be specified.

(4) It is required to judge whether the operation result is positive or negative by use of the program.

Execution Conditions

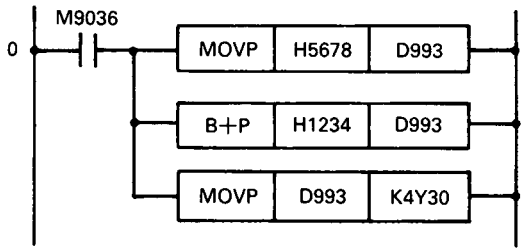


6

Program Examples

B+

Program which performs the addition of BCD data 5678 and 1234, and stores the result to D993, and at the same time outputs it to Y30 to 3F.



5678 is stored into D993 in BCD

BCD data 1234 and D993 are added and the result is stored into D993.

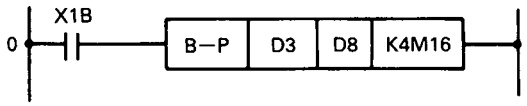
Data of D993 is output to the Y30 to 3F.

Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	MOVP	H5678	D993		
6	B+P	H1234	D993		
13	MOVP	D993	K4Y30		
18	END				

B−

Program which performs subtraction of the BCD data of D3 and that of D8 and transfers the result to M16 to 31 when X1B turns on.



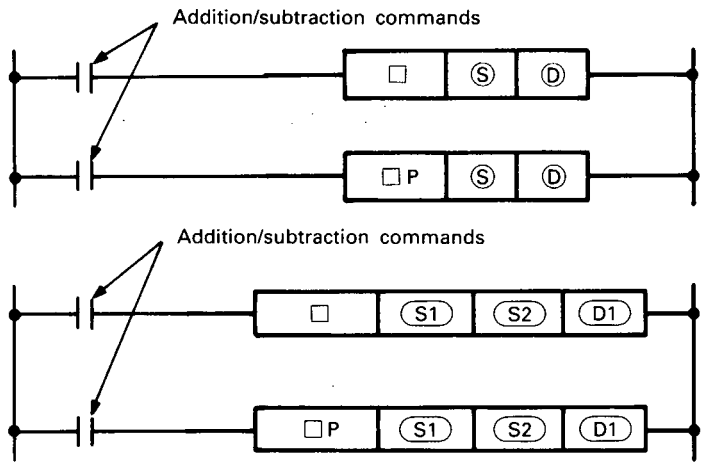
Coding

Step Number	Instruction	Device			
0	LD	X1B			
1	B−P	D3	D8	K4M16	
10	END				

6.2.6 BCD 8-digit addition, subtraction (DB+, DB+P, DB−, DB−P)

Processing Unit	Applicable CPU			
BCD 8 digits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○										
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○							K1	9					
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				to			○		○	○
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K8	11					
(D1)		○	○	○	○	○	○	○	○	○	○	○	○		○													



□ Indicates the instruction symbol.

DB+, DB−

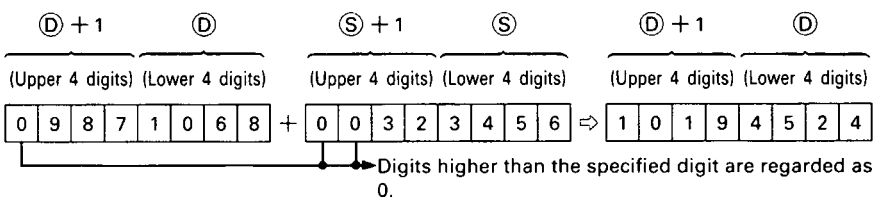
Setting data

Ⓢ	Addend/subtrahend or head device number storing addend/subtrahend
Ⓓ	Head device number storing augend/minuend
(S1)	Augend/minuend or head device number storing augend/minuend
(S2)	Addend/subtrahend or head device number storing addend/subtrahend
(D1)	Head device number which will store the operation result

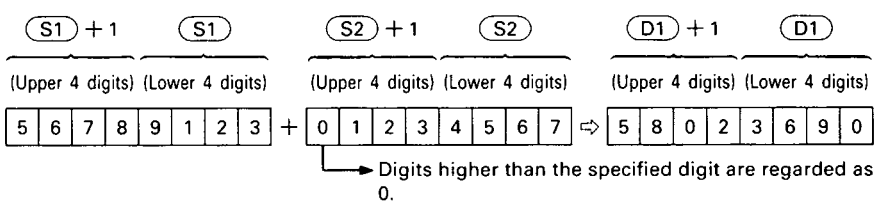
Functions

DB+

(1) Performs the addition of BCD data specified at Ⓓ and the BCD data specified at Ⓢ, and stores the addition result into the device specified at Ⓓ.



(2) Performs the addition of BCD data specified at (S1) and the BCD data specified at (S2), and stores the addition result into the device specified at (D1).

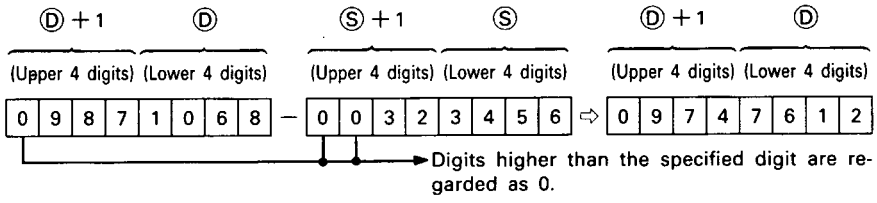


(3) At Ⓢ, (S1), (S2) and Ⓓ, 0 to 99999999 (BCD 8 digits) can be specified.

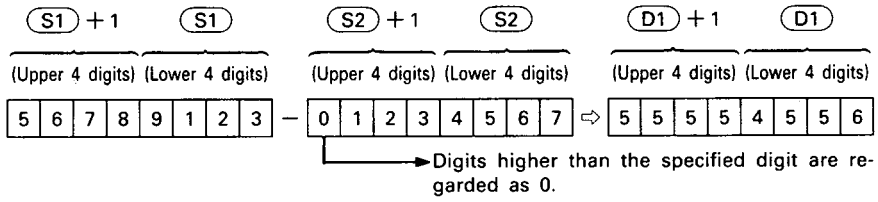
(4) Even if the addition result exceeds 99999999, the carry flag does not turn on and the carry digit is ignored.

DB—

(1) Subtracts the BCD data specified at Ⓢ from the BCD data specified at Ⓓ, and stores the subtraction result into the device specified at Ⓓ.



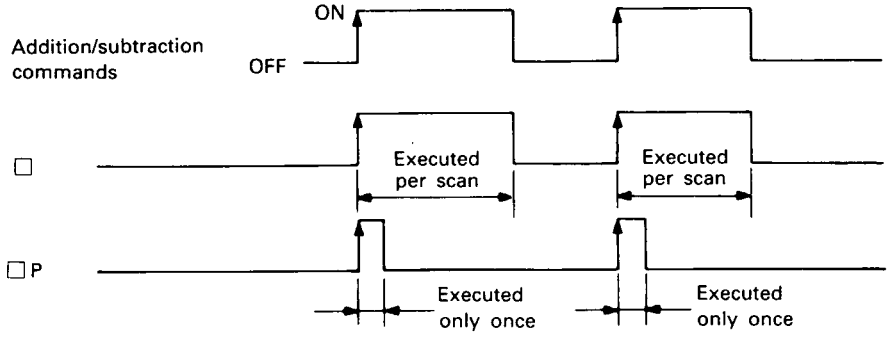
(2) Performs subtraction of the BCD data specified at (S1) and the BCD data specified at (S2), and stores the subtraction result into the device specified at (D1).



(3) At Ⓢ, (S1), (S2) and Ⓓ, 0 to 99999999 (BCD 8 digits) can be specified.

(4) It is required to judge whether the operation result is positive or negative by use of the program.

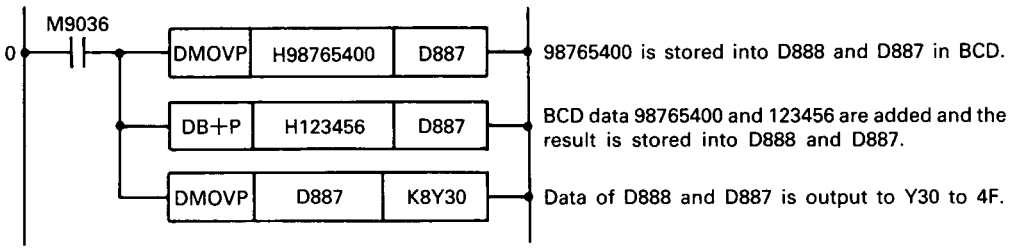
Execution Conditions



Program Examples

DB+

Program which performs the addition of BCD data 98765400 and 123456, and stores the result to D888 and D887, and at the same time, outputs it to Y30 to 4F.



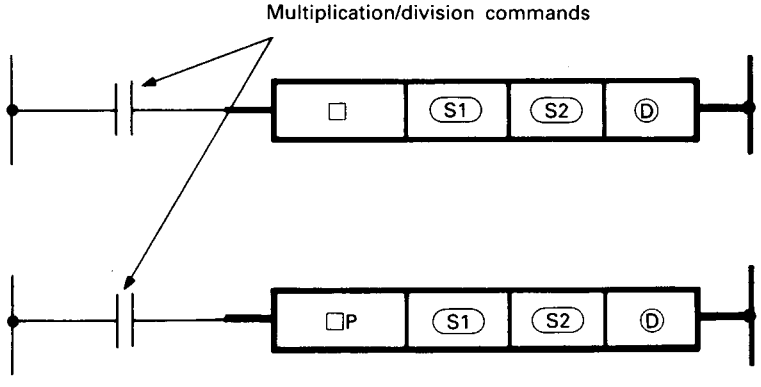
Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	DMOVP	H98765400	D887		
8	DB+P	H123456	D887		
17	DMOVP	D887	K8Y30		
24	END				

6.2.7 BCD 4-digit multiplication, division (B✳, B✳P, B/, B/P)

Processing Unit	Applicable CPU			
BCD 4 digits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	9		○		○	○
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
(D)		○	○	○	○	○	○	○	○	○	○	○	○		○							K1 to K8						



□ Indicates the instruction symbol.

B✳, B/

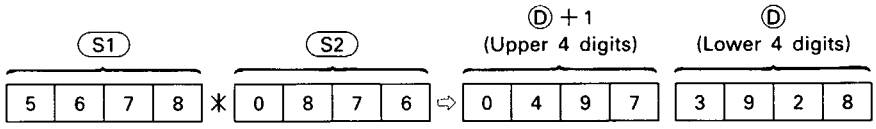
Setting data

(S1)	Multiplicand/dividend or head device number storing multiplicand/dividend
(S2)	Multiplier/divider or head device number storing multiplier/divider
(D)	Head device number which will store the result

Functions

B✳

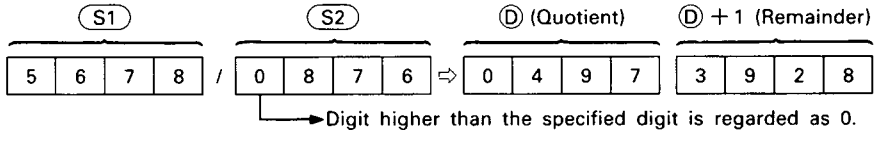
(1) Performs the multiplication of BCD data of device specified at (S1) and the BCD data of device specified at (S2), and stores the result into the device specified at (D).



(2) At (S1) and (S2), 0 to 9999 (BCD 4 digits) can be specified.

B/

(1) Performs devision of the BCD data specified at (S1) and the BCD data specified at (S2), and stores the division result into the device specified at (D).

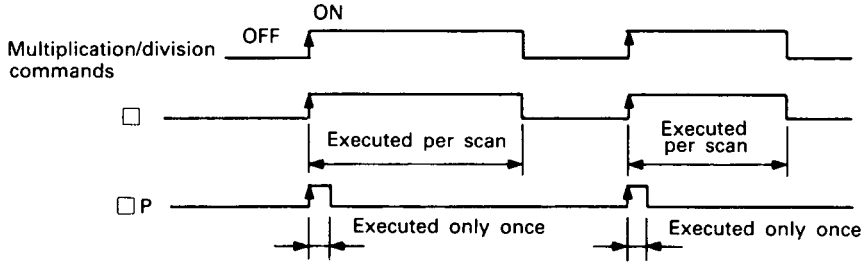


(2) In regards to the operation result, the quotient and remainder are stored by use of 32 bits.

Quotient (BCD 4 digits): Stored to the lower 16 bits.
Remainder (BCD 4 digits): Stored to the upper 16 bits.

(3) (D) will not store the remainder of the dividion result if it is a bit device.

Execution Conditions



Operation Errors

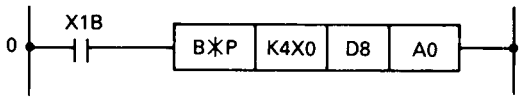
In the following cases, operation error occurs and the error flag turns on.

- A value other than 0 to 9 exists in any digit of (S1), (S2).
- The divisor (S2) is 0.

Program Examples

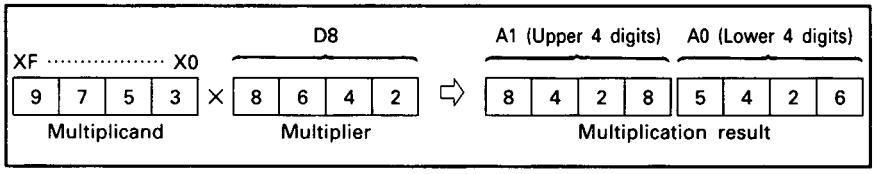
B✳

Program which performs multiplication of the BCD data of X0 to F and BCD data of D8, and stores the result into A0 and A1 when X1B turns on.



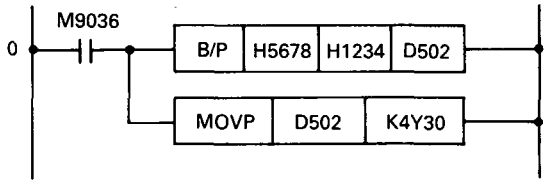
Coding

Step Number	Instruction	Device			
0	LD	X1B			
1	B✳P	K4X0	D8	A0	
10	END				



B/

Program which performs the division of BCD data 5678 and 1234, and stores the result to D502 and 503, and at the same time, outputs the quotient to Y30 to 3F.

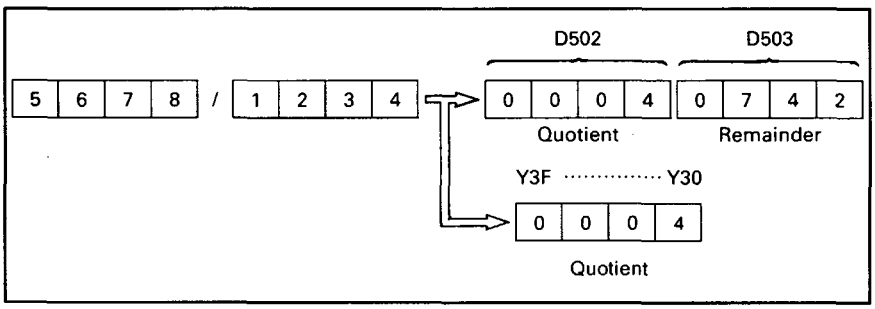


Division of BCD data 5678 and 1234 is performed and the remainder is stored into D502 and D503.

The quotient (D502) is output to the Y30 to 3F.

Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	B/P	H5678	H1234	D502	
10	MOVP	D502	K4Y30		
15	END				

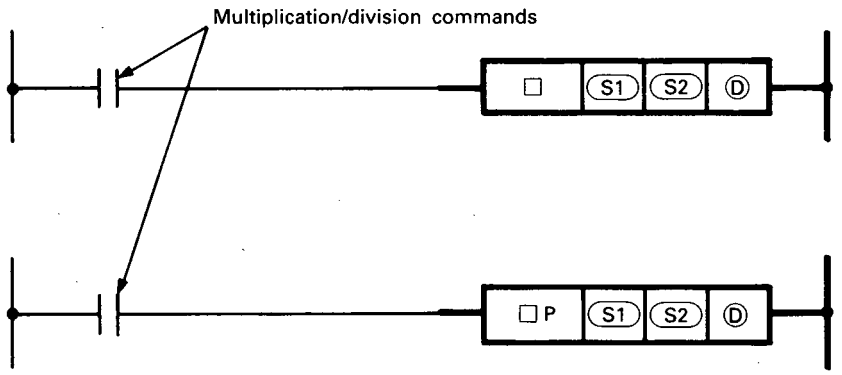


6

6.2.8 BCD 8-digit multiplication, division (DB✳, DB✳P, DB/, DB/P)

Processing Unit	Applicable CPU			
BCD 8 digits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1						
(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				to	11		○		○	○
(D)		○	○	○	○	○	○	○	○	○	○	○										K8						



□ Indicates the instruction symbol.

DB✳, DB/

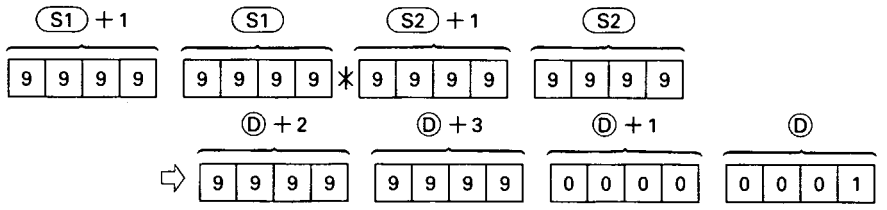
Setting data

(S1)	Multiplicand/dividend or head device number storing multiplicand/dividend
(S2)	Multiplier/divider or head device number storing multiplier/divider
(D)	Head device number which will store the result

Functions

DB✳

(1) Performs multiplication of the BCD data specified at (S1) and the BCD data specified at (S2), and stores the multiplication result into the device specified at (D).

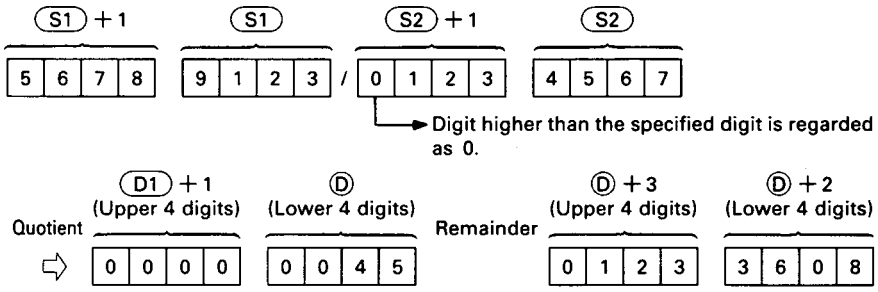


(2) If (D) is a bit device, the 8 lower digits (32 lower bits) of the multiplication result may only be specified.
K1 1 lower digit (B0 to 3), K4 4 lower digits (B0 to 15), K8 8 lower digits (B0 to 31)

(3) At (S1) and (S2), 0 to 99999999 (BCD 8 digits) can be specified.

DB/

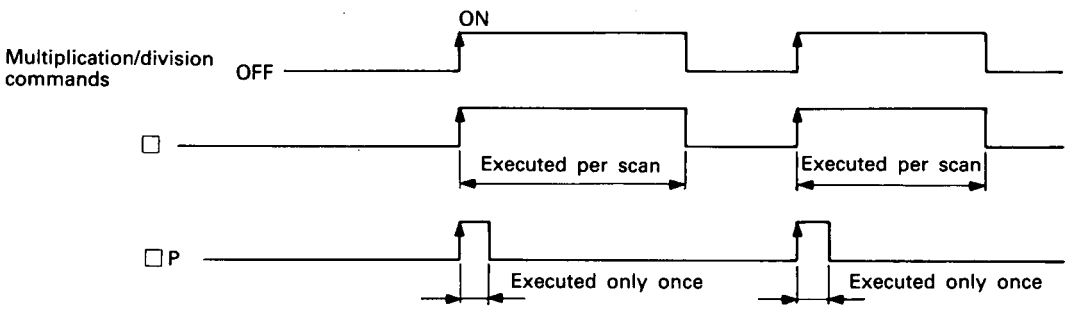
(1) Performs division of the BCD data specified at (S1) and the BCD data specified at (S2), and stores the division result into the device specified at Ⓓ.



(2) In regards to the operation result, the quotient and remainder are stored by use of 64 bits.
Quotient (BCD 8 digits): Stored to the lower 32 bits.
Remainder (BCD 8 digits): Stored to the upper 32 bits.

(3) Ⓓ will not store the remainder of the division result if it is a bit device.

Execution Conditions



Operation Errors

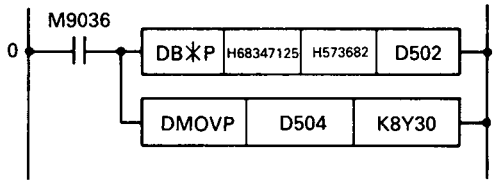
In the following cases, operation error occurs and the error flag turns on.

- A value other than 0 to 9 exists in any digit of (S1), (S2).
- The divisor (S2) is 0.

Program Examples

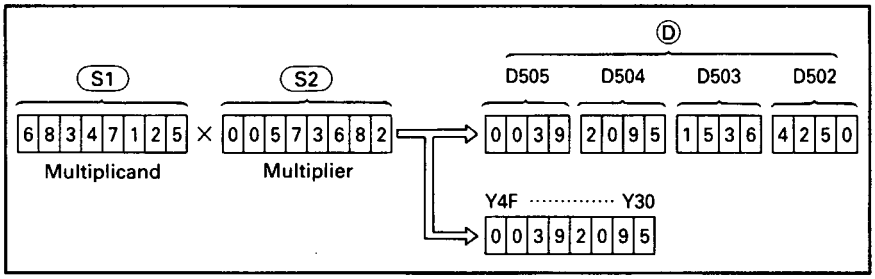
DB✳

Program which performs multiplication of the BCD data 68347125 and 573682, and stores the result to D505 to 502, and at the same time, outputs the upper 8 digits to Y30 to 4F.



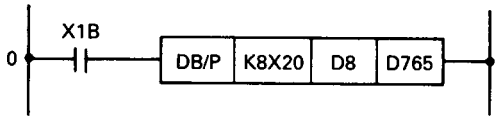
Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	DB✳P	H68347125	H573682	D502	
12	DMOVP	D504	K8Y30		
19	END				



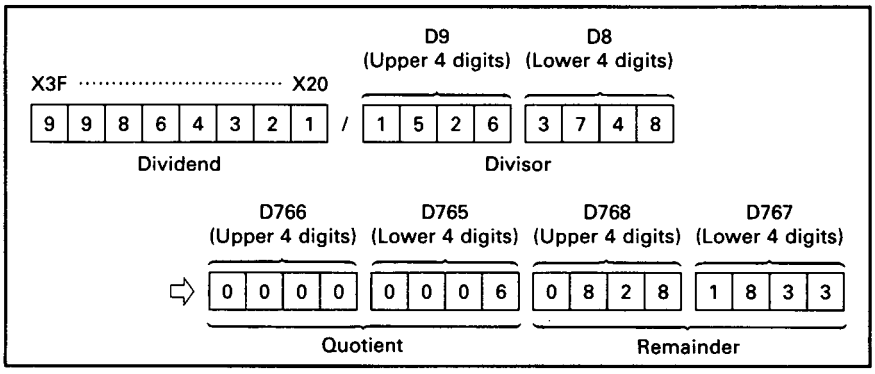
DB/

Program which performs division of the BCD data of X20 to 3F and the BCD data of D8 and 9, and stores the result to D765 to 768 when X1B turns on.



Coding

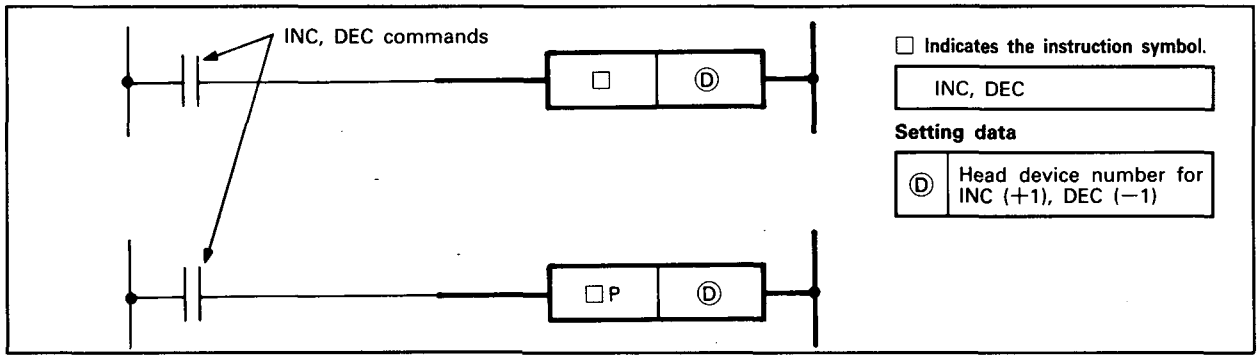
Step Number	Instruction	Device			
0	LD	X1B			
1	DB/P	K8X20	D8	D765	
12	END				



6.2.9 16-bit BIN data increment, decrement (INC, INCP, DEC, DECP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

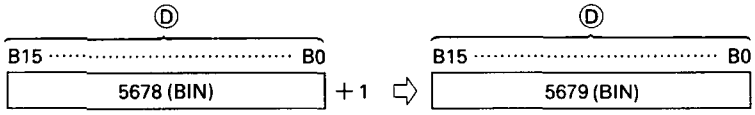
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to Δ	3	○	○		○	○



Functions

INC

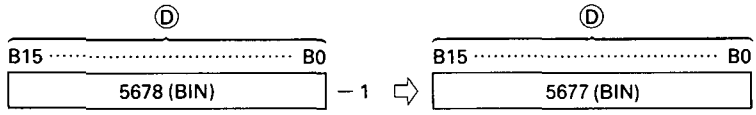
(1) Performs the addition of 1 to the device (16-bit data) specified at Ⓓ.



(2) If INC or INCP is executed when the content of device specified at Ⓓ is 32767, −32768 is stored into the device specified at Ⓓ.

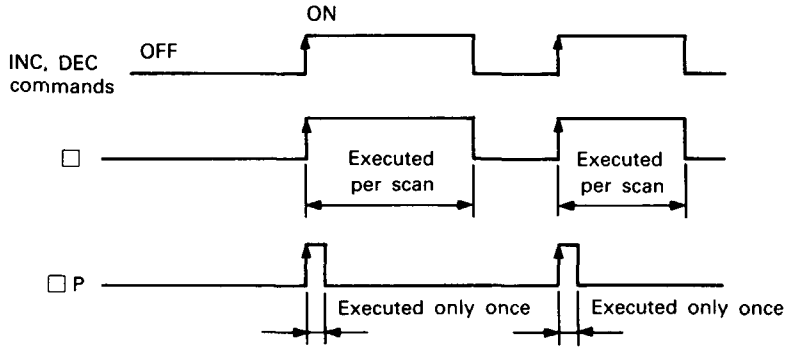
DEC

(1) Performs the subtraction of 1 from the device (16-bit data) specified at Ⓓ.



(2) If DEC or DECP is executed when the content of device specified at Ⓓ is 0, −1 is stored into the device specified at Ⓓ.

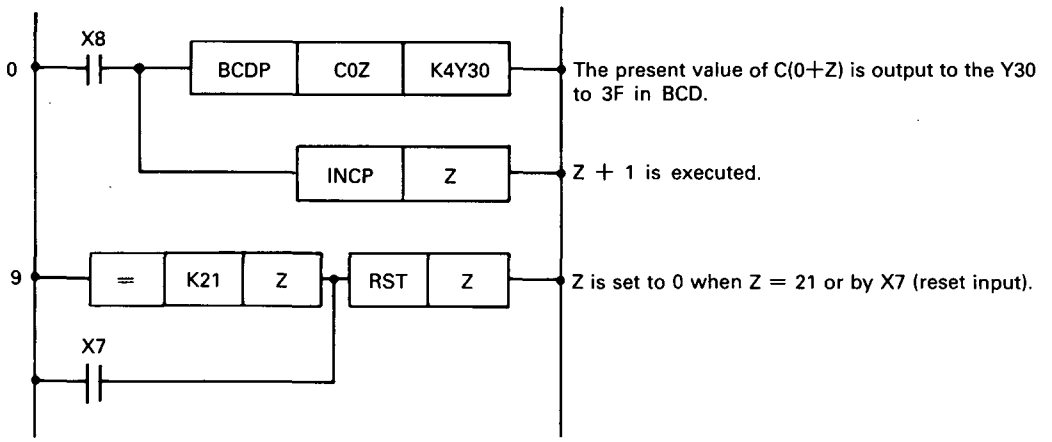
Execution Conditions



Program Examples

INC

Program which outputs the present value of counters C0 to C20 in BCD to Y30 to 3F each time X8 turns on.
(When the present value < 9999)

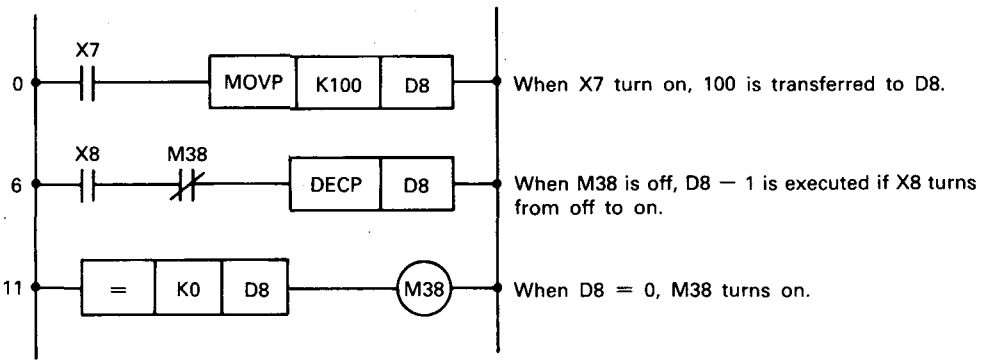


Coding

Step Number	Instruction	Device			
0	LD	X8			
1	BCDP	C0Z	K4Y30		
6	INCP	Z			
9	LD=	K21	Z		
14	OR	X7			
15	RST	Z			
18	END				

DEC

Down counter program.



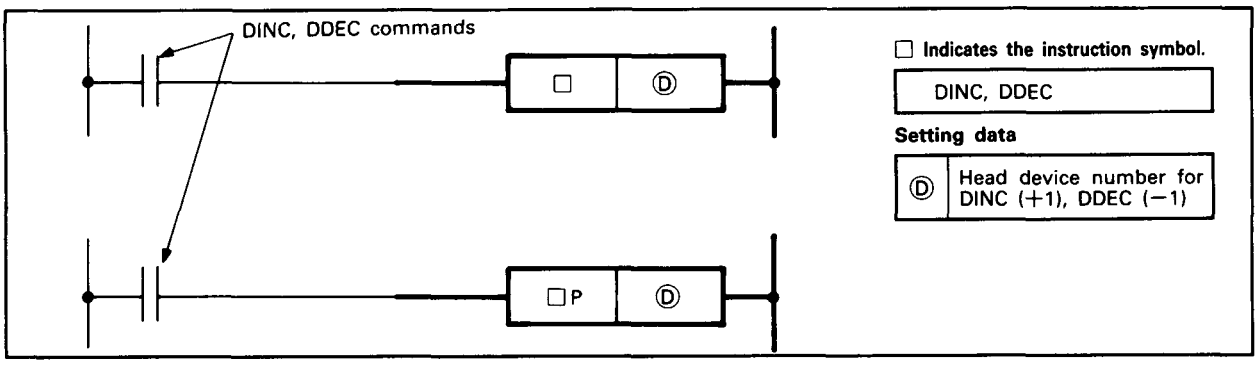
Coding

Step Number	Instruction	Device			
0	LD	X7			
1	MOVP	K100	D8		
6	LD	X8			
7	ANI	M38			
8	DECP	D8			
11	LD=	K0	D8		
16	OUT	M38			
17	END				

6.2.10 32-bit BIN data increment, decrement (DINC, DINCP, DDEC, DDECP)

Processing Unit	Applicable CPU			
32 bits	A1N	A2N	A3N	A3H

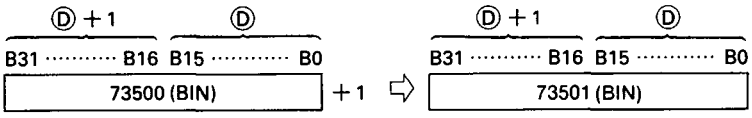
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○							K1 to K8	3	○	○		○	○



Functions

DINC

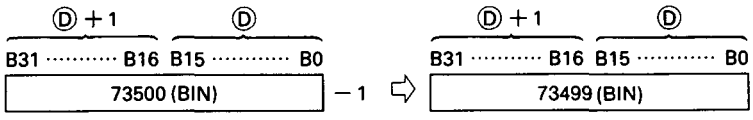
(1) Performs the addition of 1 to the device (32-bit data) specified at Ⓓ.



(2) If DINC or DINCP is executed when the content of device specified at Ⓓ is 2147483647, −2147483648 is stored into the device specified at Ⓓ.

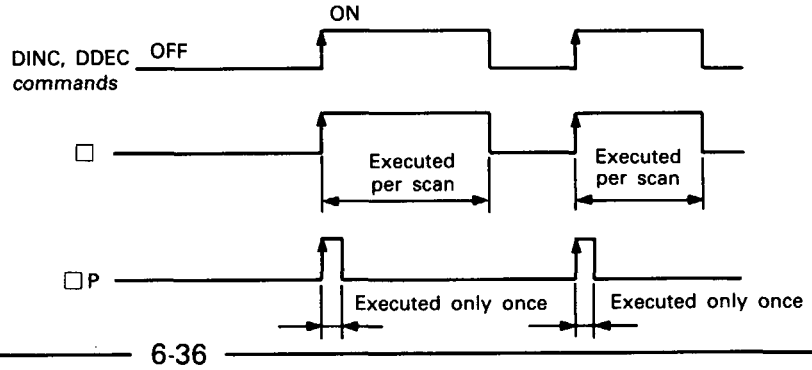
DDEC

(1) Performs the subtraction of 1 from the device (32-bit data) specified at Ⓓ.



(2) If DDEC or DDECP is executed when the content of device specified at Ⓓ is 0, −1 is stored into the device specified at Ⓓ.

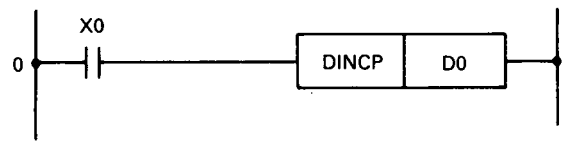
Execution Conditions



Program Examples

DINC

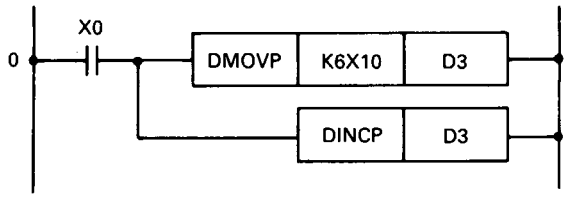
1) Program which adds 1 to the data of D0 and 1 when X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	DINCP	D0			
4	END				

2) Program which adds 1 to the data of X10 to 27 and stores the result to D3 and 4 when X0 turns on.

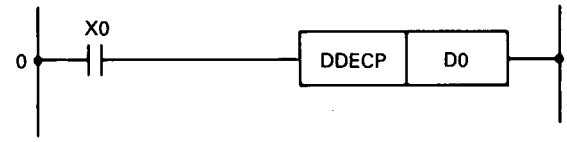


Coding

Step Number	Instruction	Device			
0	LD	X0			
1	DMOVP	K6X10	D3		
8	DINCP	D3			
11	END				

DDEC

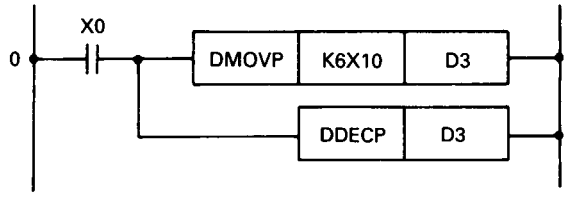
1) Program which subtracts 1 from the data of D0 and 1 when X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	DDECP	D0			
4	END				

2) Program which subtracts 1 from the data of X10 to 27 and stores the result to D3 and 4 when X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	DMOVP	K6X10	D3		
8	DDECP	D3			
11	END				

MEMO

6.3 BCD ⟷ BIN Conversion Instructions

The BCD ⟷ BIN conversion instructions are instructions which convert BCD data to BIN data and BIN data to BCD data.

Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page
BCD	BCD	6-39 to 6-41	BIN	BIN	6-42 to 6-44
	BCDP	6-39 to 6-41		BINP	6-42 to 6-44
	DBCD	6-39 to 6-41		DBIN	6-42 to 6-44
	DBCDP	6-39 to 6-41		DBINP	6-42 to 6-44

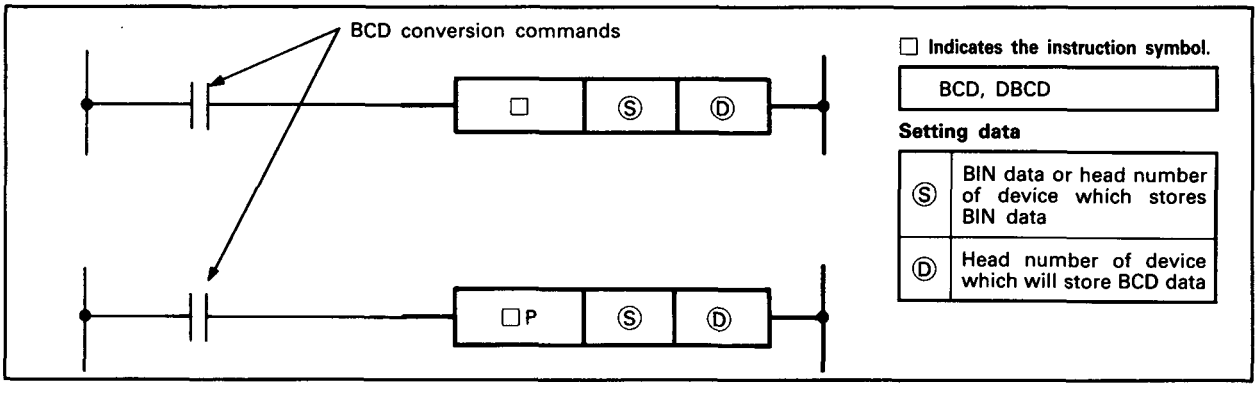
Numeric values usable for the BCD ⟷ BIN conversion instructions are as follows:

BCD, BCDP, BIN, BINP: 0 to 9999
DBCD, DBCDP, DBIN, DBINP: 0 to 99999999

6.3.1 BIN data → BCD 4-, 8-digit conversion (BCD, BCDP, DBCD, DBCDP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
BCD	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4	5	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DBCD	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○							K1 to K8	9					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



□ Indicates the instruction symbol.

BCD, DBCD

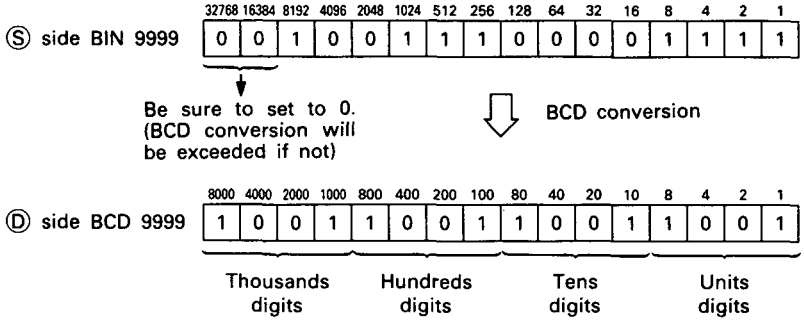
Setting data

Ⓢ	BIN data or head number of device which stores BIN data
Ⓓ	Head number of device which will store BCD data

Functions

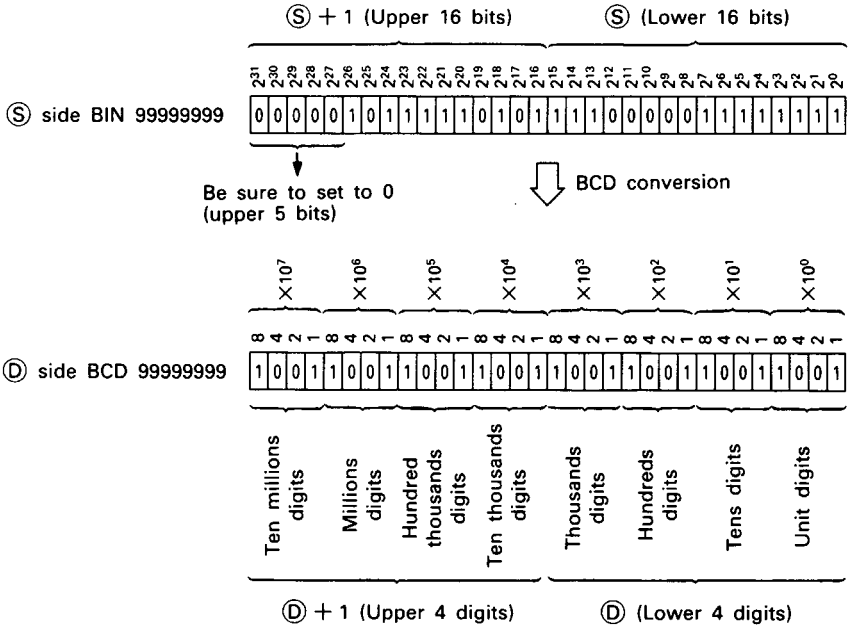
BCD

Converts BIN data (0 to 9999) of the device specified at Ⓢ into BCD and transfers the result to the device specified at Ⓓ.

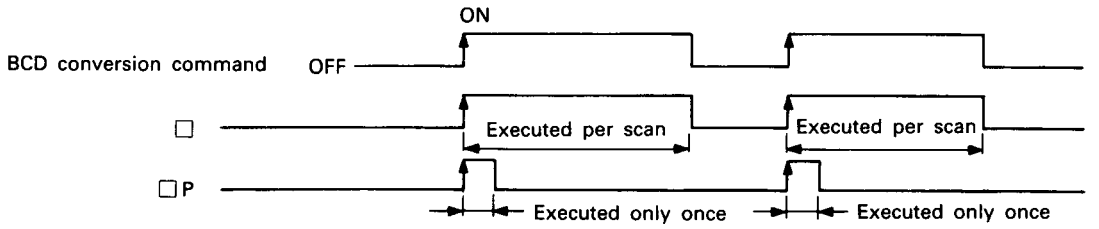


DBCD

Converts BIN data (0 to 99999999) of the device specified at Ⓢ into BCD and transfers the result to the device specified at Ⓓ.



Execution Conditions



Operation Errors

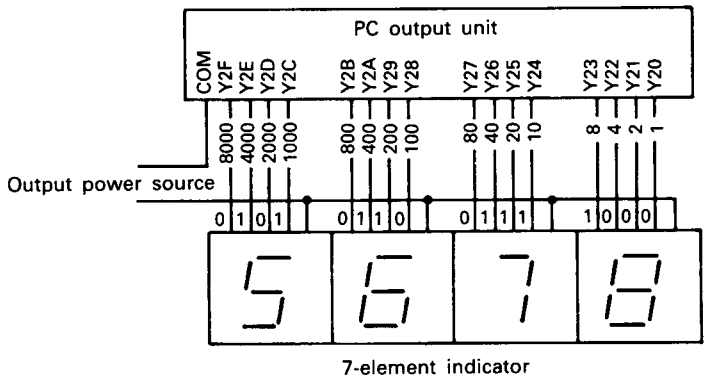
In the following case, operation error occurs and the error flag turns on.

- When BCD instruction is used
  The data of source Ⓢ is outside the limits of 0 to 9999.
- When DBCD instruction is used
  The data of source Ⓢ is outside the range of 0 to 99999999.

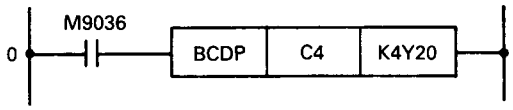
Program Examples

BCD

Program which outputs the present value of C4 from the Y20 to 2F to the BCD indicator.



7-element indicator

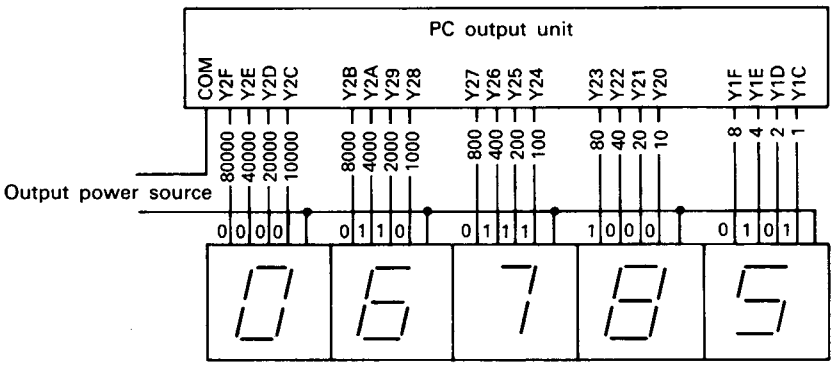


Coding

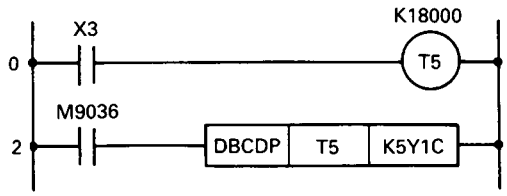
Step Number	Instruction	Device			
0	LD	M9036			
1	BCDP	C4	K4Y20		
6	END				

DBCD

Program which outputs the present value of timer, of which set value exceeds 9999, to the Y1C to 2F.



7-element indicator



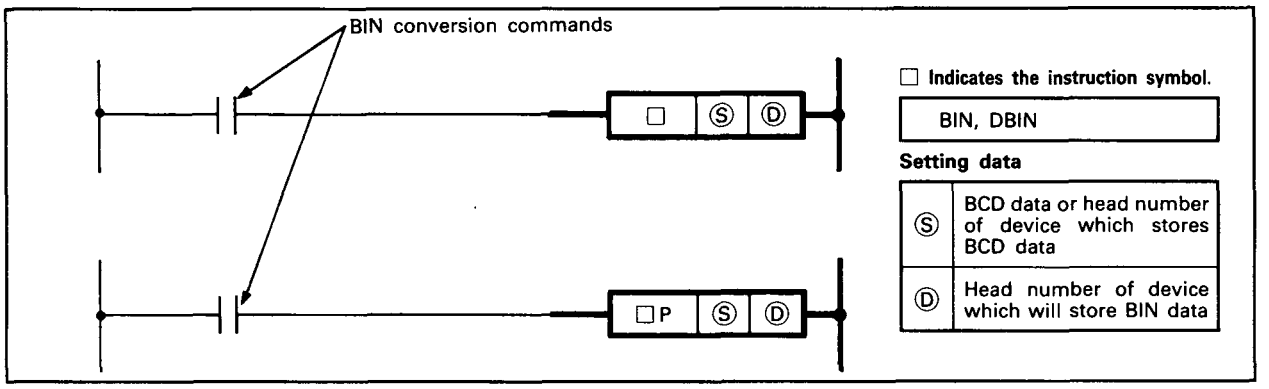
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	OUT	T5	K18000		
2	LD	M9036			
3	DBCDP	T5	K5Y1C		
12	END				

6.3.2 BCD 4-, 8-digit → BIN data conversion (BIN, BINP, DBIN, DBINP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

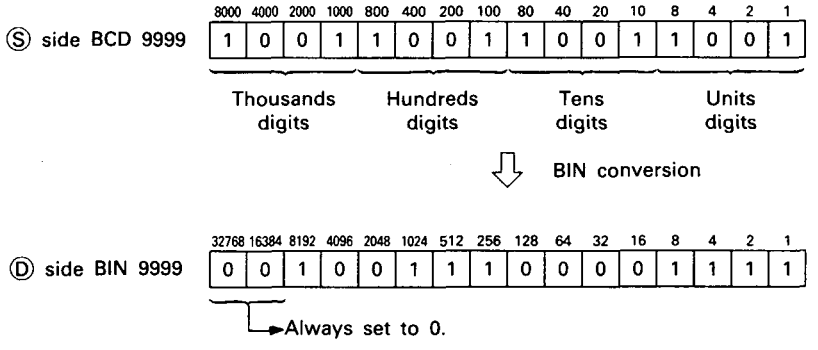
		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
BID	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4	5	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DBIN	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○							K1 to K8	9					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



Functions

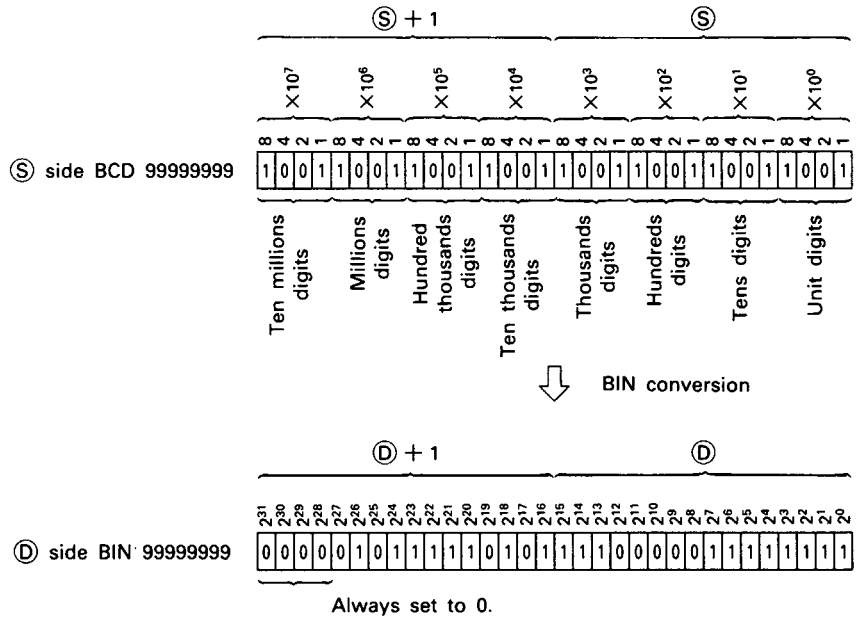
BIN

Converts BCD data (0 to 9999) of device specified at Ⓢ into BIN and transfers the result to the device specified at Ⓓ.

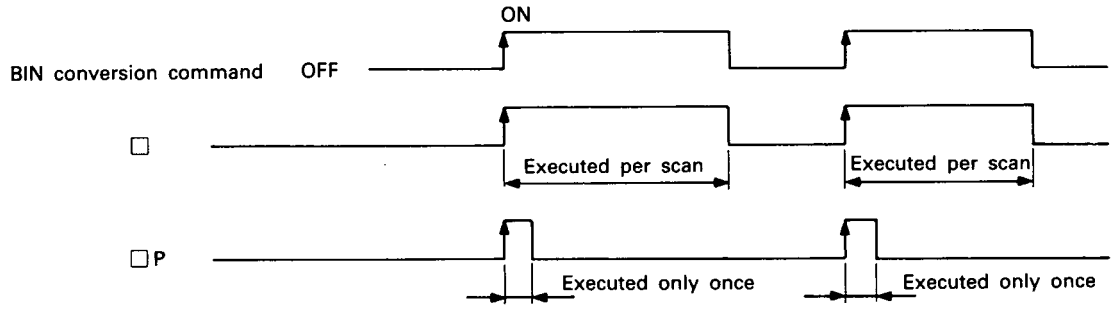


DBIN

Converts BCD data (p to 99999999) of device specified at Ⓢ into BIN and transfers the result to the device specified at Ⓓ.



Execution Conditions



Operation Error

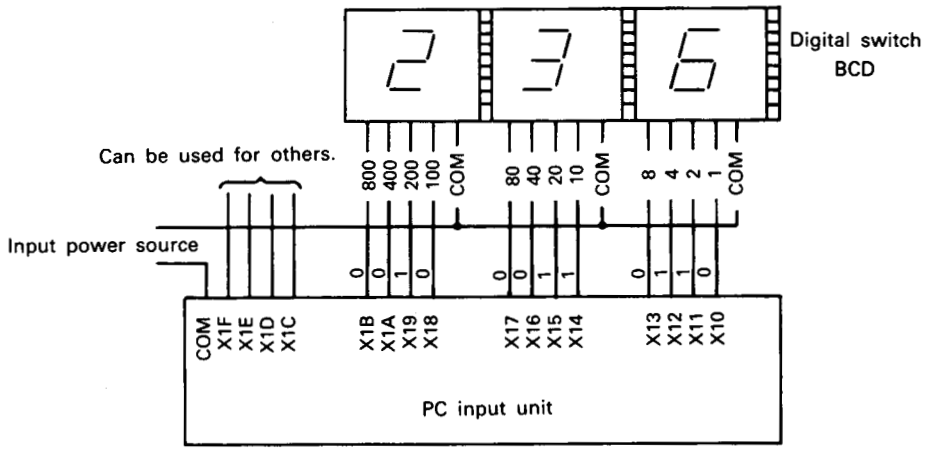
In the following case, operation error occurs and the error flag turns on.

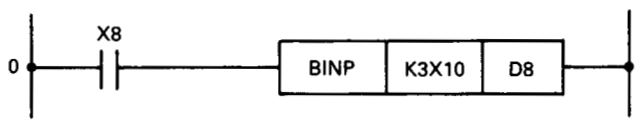
○ Each digit of source Ⓢ is outside the range of 0 to 9.

Program Examples

BIN

Program which converts the BCD data of X10 to 1B into BIN and stores the result into D8 when X8 turns on.



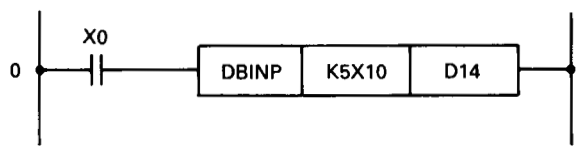


Coding

Step Number	Instruction	Device			
0	LD	X8			
1	BINP	K3X10	D8		
6	END				

DBIN

Program which converts the BCD data of X10 to 23 into BIN and stores the result into D14 and 15 when X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	DBINP	K5X10	D14		
10	END				

MEMO

6.4 Data Transfer Instructions

The data transfer instructions are instructions which perform data transfer, interchanging data, the negative (reverse) data transfer, etc.

Classification	Instruction Symbol	Ref. Page
Transfer	MOV	6-46 to 6-47
	MOVP	6-46 to 6-47
	DMOV	6-46 to 6-47
	DMOVP	6-46 to 6-47
Negative transfer	CML	6-48 to 6-50
	CMLP	6-48 to 6-50
	DCML	6-48 to 6-50
	DCMLP	6-48 to 6-50
Block transfer	BMOV	6-51 to 6-53
	BMOVP	6-51 to 6-53
Same data block transfer	FMOV	6-51 to 6-53
	FMOVP	6-51 to 6-53
Interchange	XCH	6-54 to 6-55
	XCHP	6-54 to 6-55
	DXCH	6-54 to 6-55
	DXCHP	6-54 to 6-55

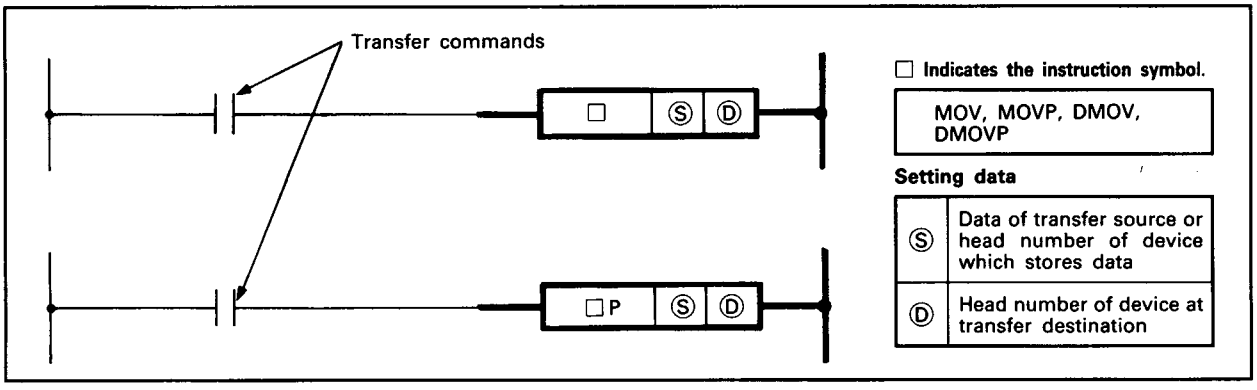
POINT

The data moved by the data transfer instruction (transfer, interchanging, negative transfer, block transfer, block transfer of the same data) is retained until new data is transferred. Therefore, even if the execution command of each instruction turns off, the data does not change.

6.4.1 16-, 32-bit data transfer (MOV, MOVP, DMOV, DMOVP)

Processing Unit	Applicable CPU			
12/32 bits	A1N	A2N	A3N	A3H

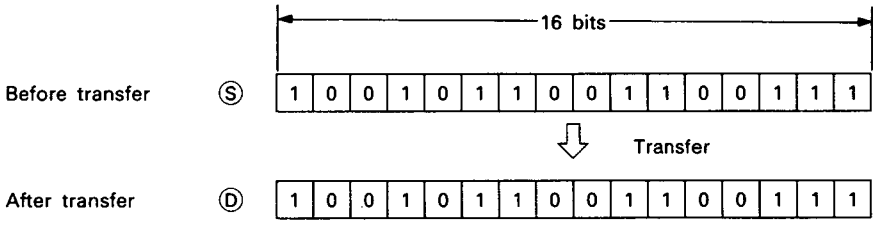
		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
MOV	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DMOV	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1 to K8	7					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



Functions

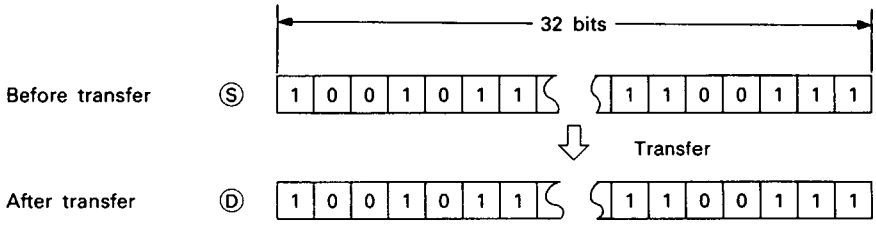
MOV

Transfers the 16-bit data of the device specified at Ⓢ to the device specified at Ⓓ.



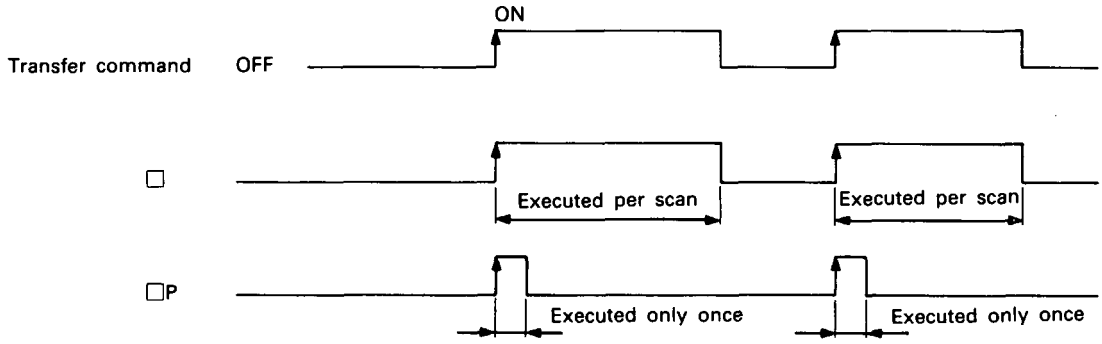
DMOV

Transfers the 32-bit data of the device specified at Ⓢ to the device specified at Ⓓ.



6

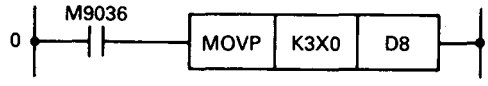
Execution Conditions



Programs Examples

MOV

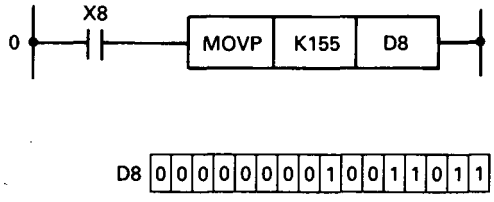
1) Program which stores the data of inputs X0 to B into D8.



Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	MOVP	K3X0	D8		
6	END				

2) Program which stores 155 into D8 as a binary value when X8 turns on.

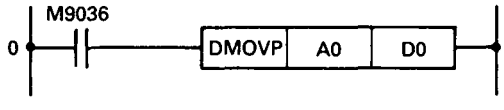


Coding

Step Number	Instruction	Device			
0	LD	X8			
1	MOVP	K155	D8		
6	END				

DMOV

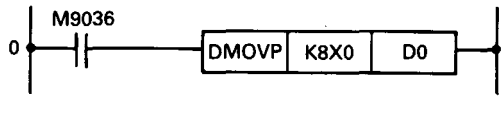
1) Program which stores the data of A0 and A1 into D0 and D1.



Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	DMOVP	A0	D0		
8	END				

2) Program which stores the data of X0 to 1F into D0 and D1.



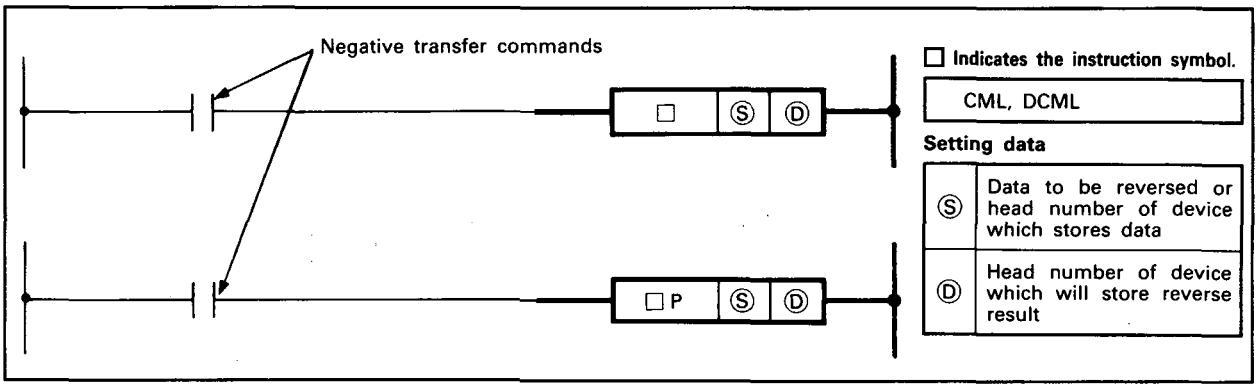
Coding

Step Number	Instruction	Device			
0	LD	M9036			
1	DMOVP	K8X0	D0		
8	END				

6.4.2 16-, 32-bit data negation transfer (CML, CMLP, DCML, DCMLP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

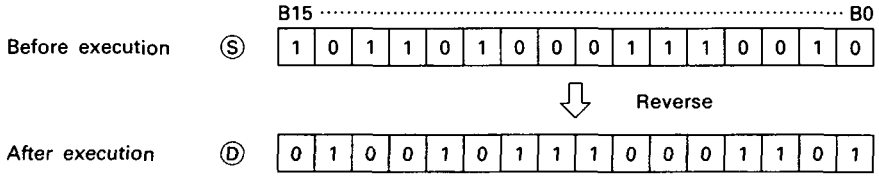
		Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level						
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
CML	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DCML	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1 to K8	7					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



Functions

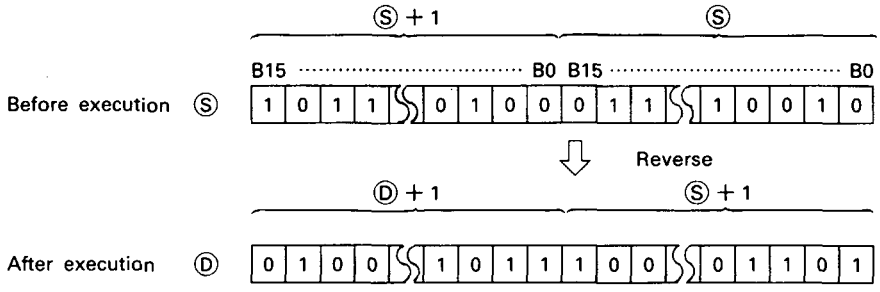
CML

Reverses the 16-bit data of Ⓢ per bit and transfers the result to Ⓓ.

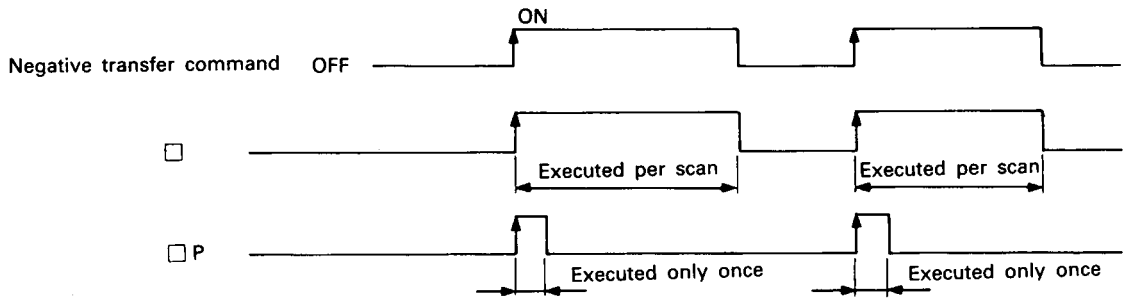


DCML

Reverses the 32-bit data of Ⓢ per bit and transfers the result to Ⓓ.



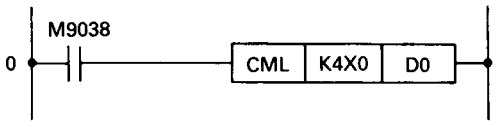
Execution Conditions



Program Examples

CML

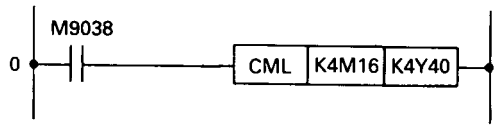
1) Program which reverses the data of X0 to F and transfers the result to D0.



Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	CML	K4X0	D0		
6	END				

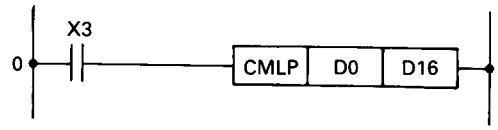
2) Program which reverses the data of M16 to 31 and transfers the result to the Y40 to 4F.



Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	CML	K4M16	K4Y40		
6	END				

3) Program which reverses the data of D0 and stores the result to D16 when X3 turns on.

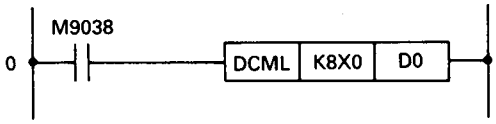


Coding

Step Number	Instruction	Device			
0	LD	X3			
1	CMLP	D0	D16		
6	END				

DCML

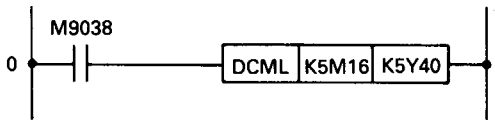
1) Program which reverses the data of X0 to 1F and transfers the result to D0 and 1.



Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	DCML	K8X0	D0		
8	END				

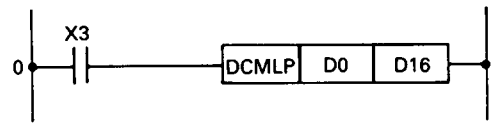
2) Program which reverses the data of M16 to 35 and transfers the result to the Y40 to 53.



Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	DCML	K5M16	K5Y40		
8	END				

3) Program which reverses the data of D0 and 1 and stores the result to D16 and 17 when X3 turns on.



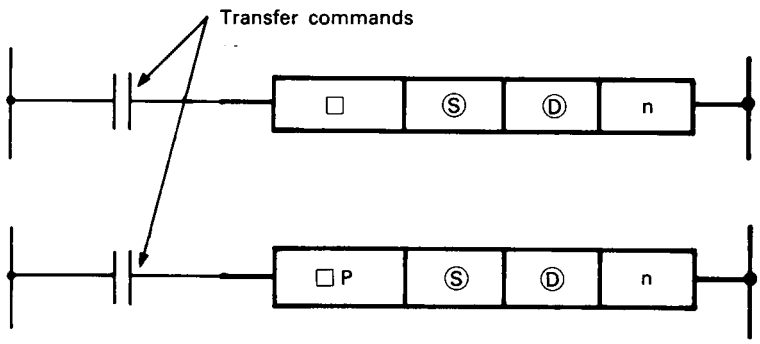
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	DCMLP	D0	D16		
8	END				

6.4.3 16-bit data block transfer (BMOV, BMOVP, FMOV, FMOVP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
BMOV	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○										K1 to K4	9		○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○																
	ⓝ																	○	○										
FMOV	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4						
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○																
	ⓝ																	○	○										



□ Indicates the instruction symbol.

BOMV, FMOV

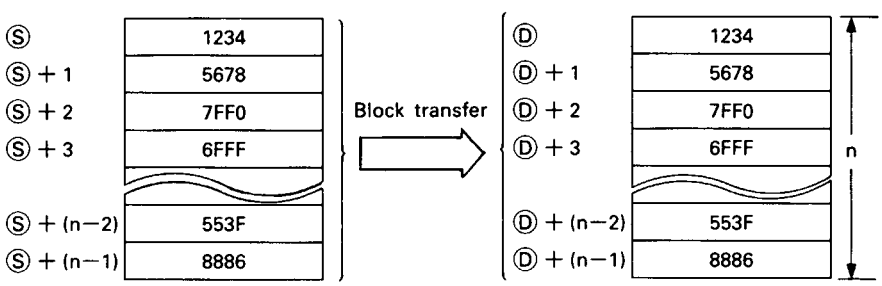
Setting data

Ⓢ	Head number of device which stores data to be transferred
Ⓓ	Head number of device which will store transferred data
n	Number of transferred points

Functions

BMOV

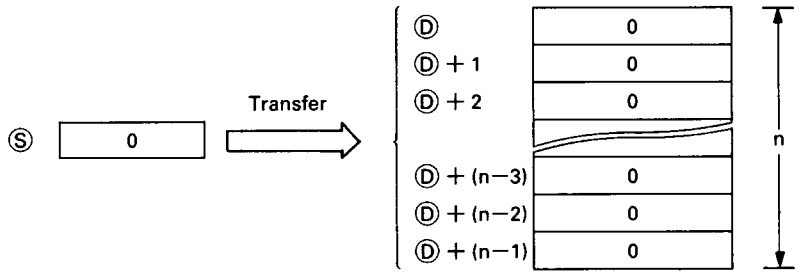
Transfers the content of "n" points, which begin with the device specified at Ⓢ, in blocks to "n" points which begin with the device specified at Ⓓ.



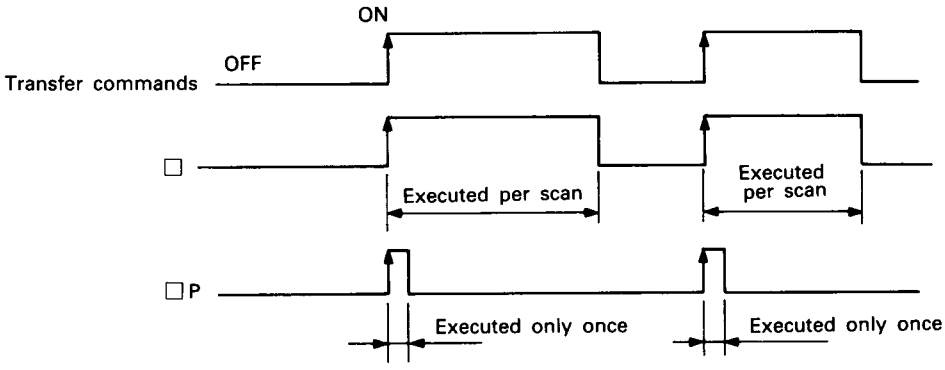
When the same devices have been specified, the transfer to the devices with the lower numbers is made in order of lower device numbers, and the transfer to the devices with the higher numbers is made in order of higher devices numbers.
The number of Ⓢ and Ⓓ digits must be equal when both Ⓢ and Ⓓ are bit devices.

FMOV

Transfers the content of device specified at Ⓢ in blocks to "n" points which begin with the device specified at Ⓓ.



Execution Conditions



Operation Error

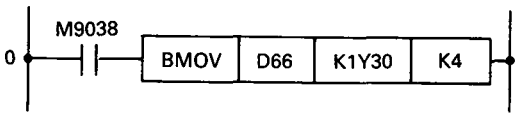
In the following case, operation error occurs and the error flag turns on.

- The transfer range exceeds the corresponding device range.

Program Examples

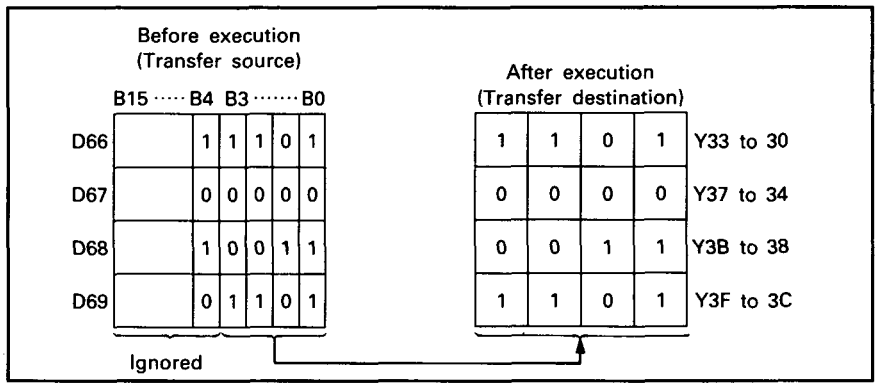
BMOV

Program which outputs the data of the lower 4 bits of D66 to 69 to the Y30 to 3F in units of 4 points.



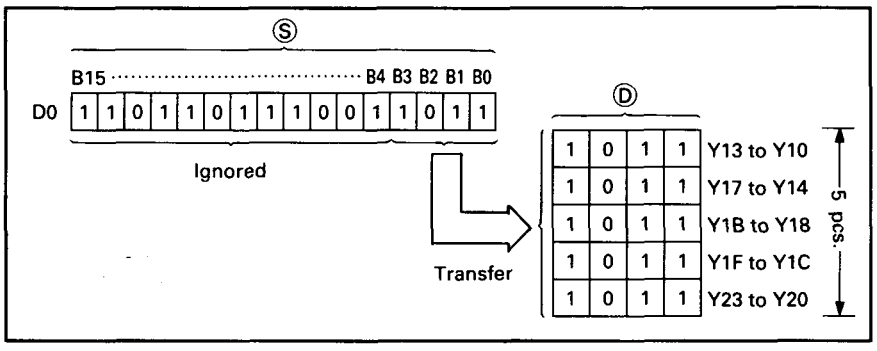
Coding

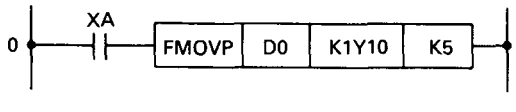
Step Number	Instruction	Device			
0	LD	M9038			
1	BMOV	D66	K1Y30	K4	
10	END				



FMOV

Program which outputs the data of the lower 4 bits of D0 to Y10 to 23 in units of 4 points when XA turns on.





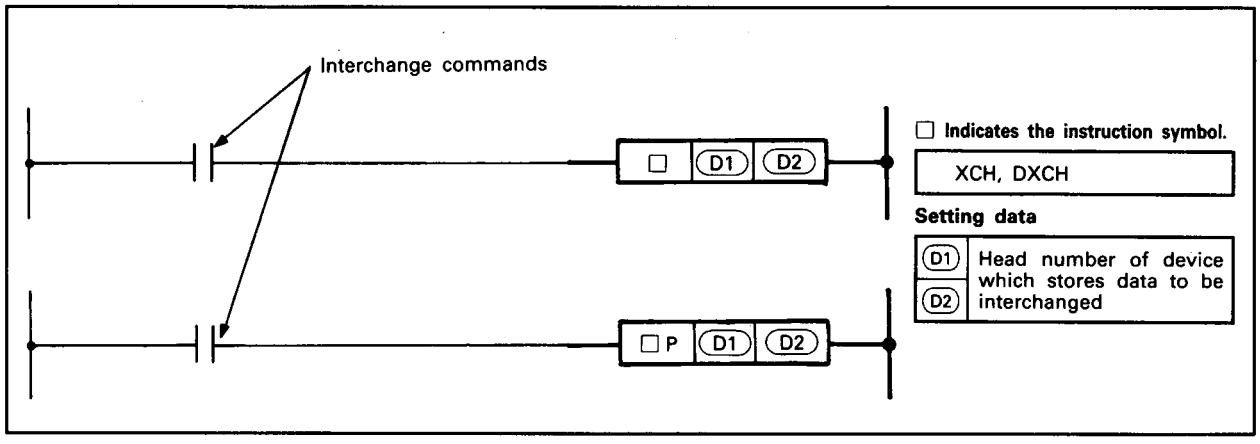
Coding

Step Number	Instruction	Device			
0	LD	XA			
1	FMOVP	D0	K1Y10	K5	
10	END				

6.4.4 16-, 32-bit data exchange (XCH, XCHP, DXCH, DXCHP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

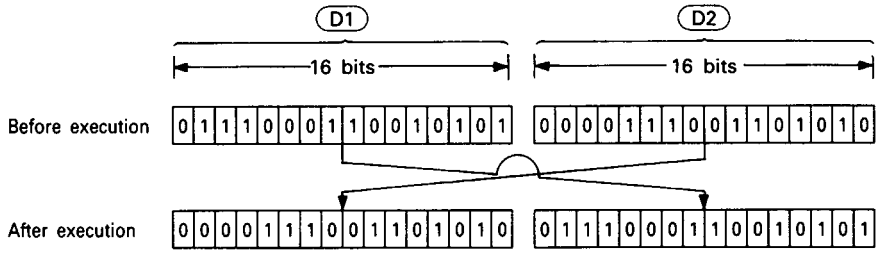
		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
XCH	(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4	5	○	○		○	○
	(D2)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DXCH	(D1)		○	○	○	○	○	○	○	○	○	○	○	○		○							K1 to K8	7					
	(D2)		○	○	○	○	○	○	○	○	○	○	○	○		○													



Functions

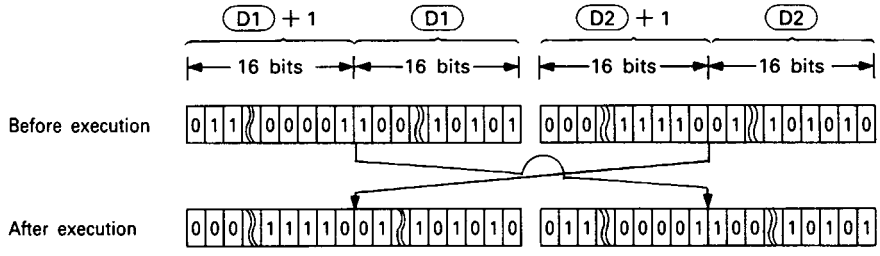
XCH

Interchanges the 16-bit data of (D1) and (D2).

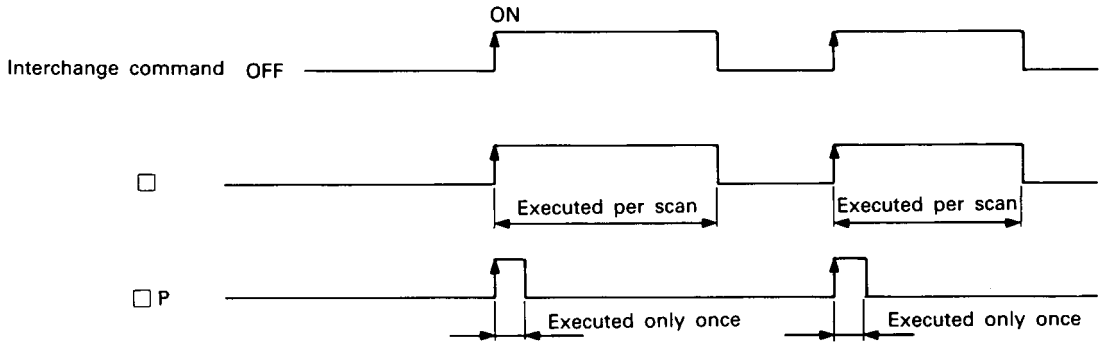


DXCH

Interchanges the 32-bit data of (D1) and (D2).



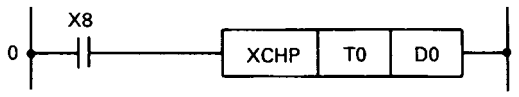
Execution Conditions



Program Examples

XCH

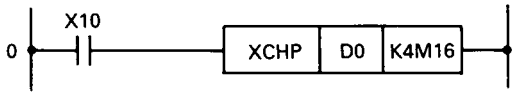
1) Program which interchanges the present value of T0 and the content of D0 when X8 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X8			
1	XCHP	T0	D0		
6	END				

2) Program which interchanges the content of D0 and the data of M16 to 31 when X10 turns on.

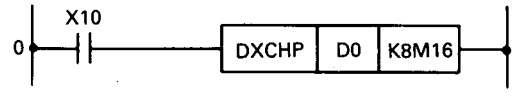


Coding

Step Number	Instruction	Device			
0	LD	X10			
1	XCHP	D0	K4M16		
6	END				

DXCH

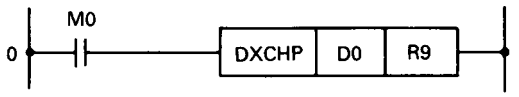
1) Program which interchanges the content of D0 and 1 and the data of M16 to 47 when X10 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X10			
1	DXCHP	D0	K8M16		
8	END				

2) Program which interchanges the content of D0 and 1 with that of R9 and 10 when M0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	M0			
1	DXCHP	D0	R9		
8	END				

7. APPLICATION INSTRUCTIONS

Application instructions are used when special processing is required. They are classified as follows:

Classification of Application Instructions	Description	Ref. Page
Logical operation instruction	Logical operation such as logical add and logical product	7-2 to 7-20
Rotation instruction	Rotation of specified data	7-21 to 7-29
Shift instruction	Shift of specified data	7-30 to 7-36
Data processing instruction	Data processing such as 16-bit data search, decode, and encode	7-37 to 7-52
FIFO instruction	Read/write of FIFO table	7-53 to 7-57
Buffer memory access instruction	Read/write of buffer memory in special function module	7-58 to 7-62
Local, remote I/O station access instruction	Read/write of data in local, remote I/O station	7-63 to 7-70
Display instruction	Output of character code, indication of data on LED display	7-71 to 7-85
Miscellaneous	Instructions which are not included in the above classification, such as WDT reset and carry flag set/reset	7-86 to 7-96

7.1 Logical Operation Instructions

(1) The logical operation instructions are instructions which perform the logical operations such as logical add and logical product.

(2) The logical operation instructions are available in the following 26 types.

Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page
Logical product	WAND	7-3 to 7-6	Exclusive OR	WXOR	7-11 to 7-14	2's complement	NEG	7-19 to 7-20
	WANDP	7-3 to 7-6		WXORP	7-11 to 7-14		NEGP	7-19 to 7-20
	DAND	7-3 to 7-6		DXOR	7-11 to 7-14			
	DANDP	7-3 to 7-6		DXORP	7-11 to 7-14			
Logical add	WOR	7-7 to 7-10	Exclusive NOR	WXNR	7-15 to 7-18			
	WORP	7-7 to 7-10		WXNRP	7-15 to 7-18			
	DOR	7-7 to 7-10		DXNR	7-15 to 7-18			
	DORP	7-7 to 7-10		DXNRP	7-15 to 7-18			

REMARKS

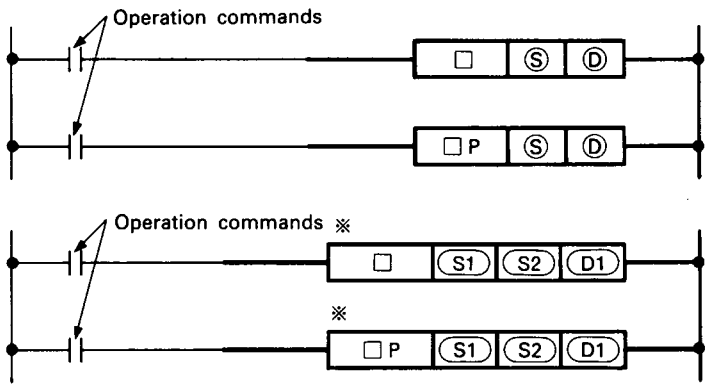
The logical operation instructions perform the following processings in units of one bit.

Classification	Processing	Operation Expression	Example A	Example B	Example Y
Logical product	Set to 1 only when both inputs A and B are 1. Set to 0 otherwise.	$Y = A \cdot B$	0	0	0
			0	1	0
			1	0	0
			1	1	1
Logical add	Set to 0 only when both inputs A and B are 0. Set to 1 to 1 otherwise.	$Y = A + B$	0	0	0
			0	1	1
			1	0	1
			1	1	1
Exclusive OR	Set to 0 when inputs A and B are equal. Set to 1 when they are different.	$Y = \overline{A} \cdot B + A \cdot \overline{B}$	0	0	0
			0	1	1
			1	0	1
			1	1	0
Exclusive NOR	Set to 1 when inputs A and B are equal. Set to 0 when they are different.	$Y = (\overline{A} + B)(A + \overline{B})$	0	0	1
			0	1	0
			1	0	0
			1	1	1

7.1.1 16-, 32-bit data logical product (WAND, WANDP, DAND, DANDP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
WAND	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5 to 7	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
	(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DAND	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1 to K8	9					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



□ Indicates the instruction symbol.

WAND, DAND

Setting data

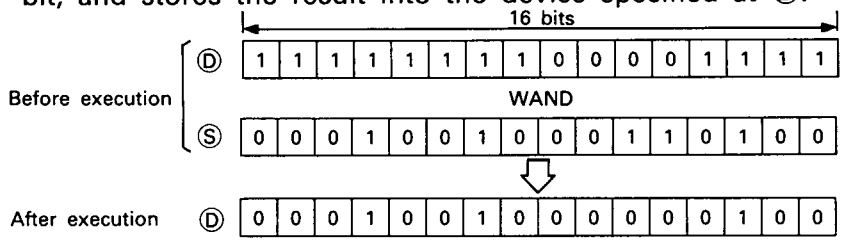
Ⓢ (S1) (S2) Ⓓ	Data for which logical product will be performed or head number of device which stores data
(D1)	Head number of device which will store the result of logical product

WAND may only be used in the areas marked ※.

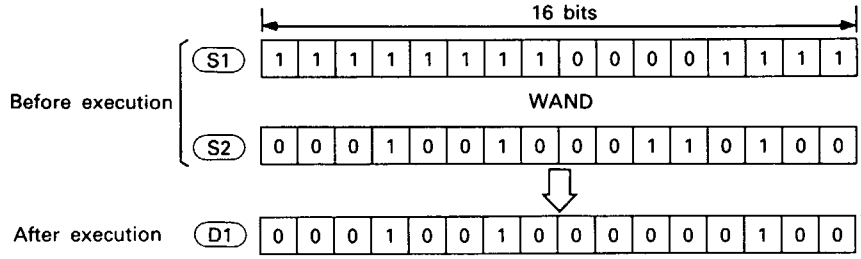
Functions

WAND

(1) Performs the logical product of the 16-bit data of device specified at Ⓓ and the 16-bit data of device specified at Ⓢ per bit, and stores the result into the device specified at Ⓓ.



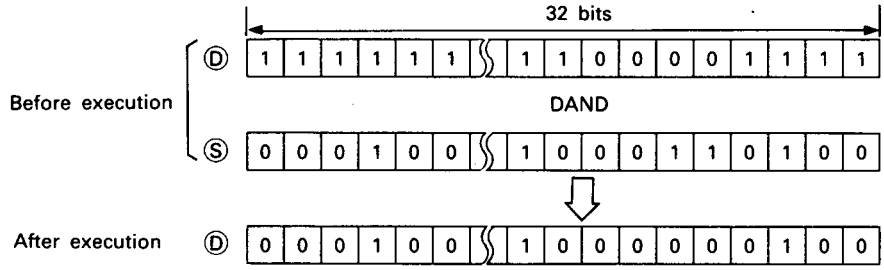
(2) Performs the logical product of the 16-bit data of device specified at (S1) and the 16-bit data of device specified at (S2) per bit, and stores the result into the device specified at (D1).



(3) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

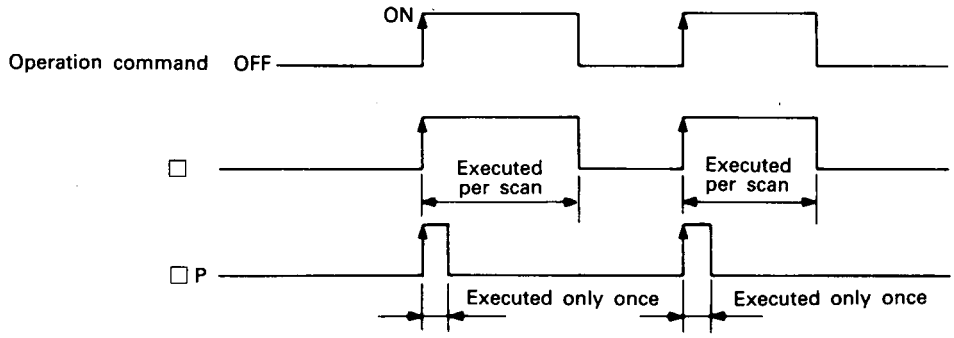
DAND

(1) Performs the logical product of the 32-bit data of device specified at Ⓓ and the 32-bit data of device specified at Ⓢ per bit, and stores the result into the device specified at Ⓓ.



(2) When operation is performed, the digits of the bit device higher than these specified are regarded as 0.

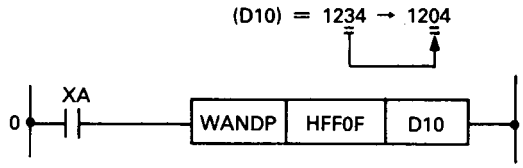
Execution Conditions



Program Examples

WAND

1) Program which masks the digit of tens (the second digit from the right), among the BCD four digits of D10, and sets it to 0 when XA turns on.



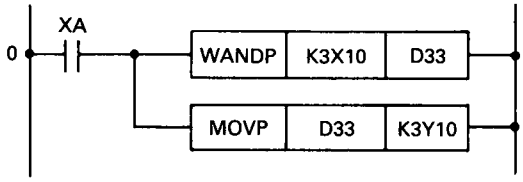
Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WANDP	HFF0F	D10		
6	END				

7

IB (NA) 66147-A

2) Program which performs logical product of the data of X10 to 1B and the data of D33, and outputs the result to the Y30 to 3B when XA turns on.

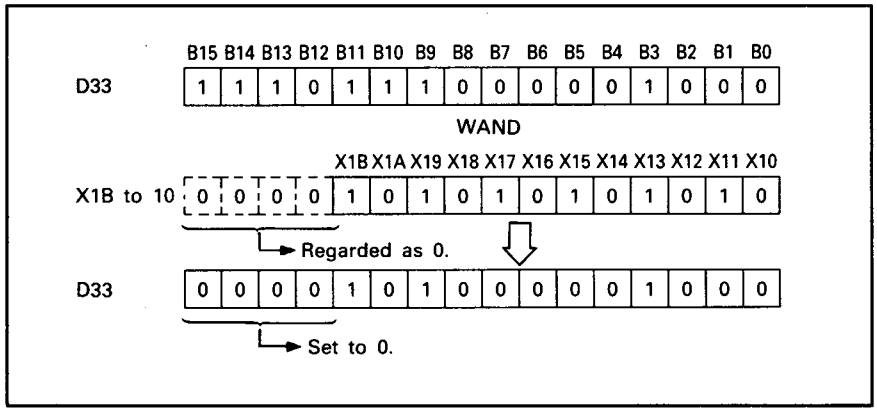


Logical product of the data of X10 to 1B and the data of D33 is performed and the result is stored into D33.

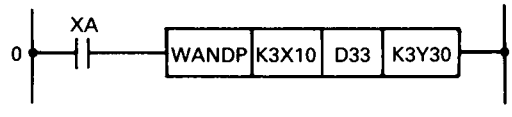
Data of D33 is output to the Y30 to 3F.

Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WANDP	K3X10	D33		
6	MOVP	D33	K3Y10		
11	END				

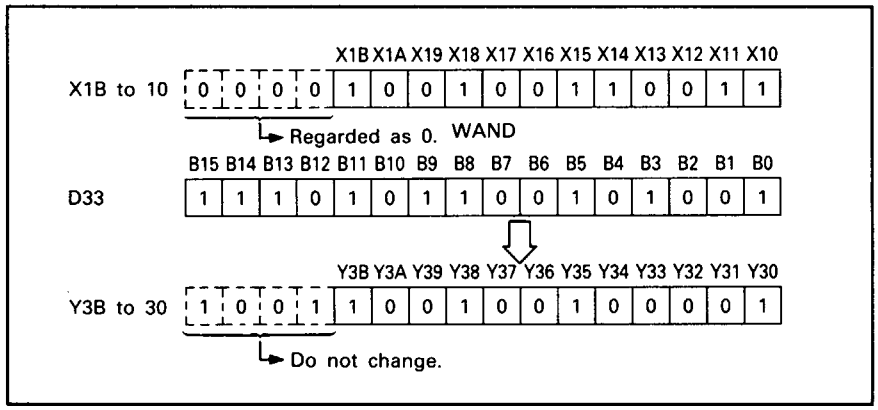


3) Program which performs logical product of the data of X10 to 17 and the data of D33, and sends the result to the Y30 to 3B when XA turns on.



Coding

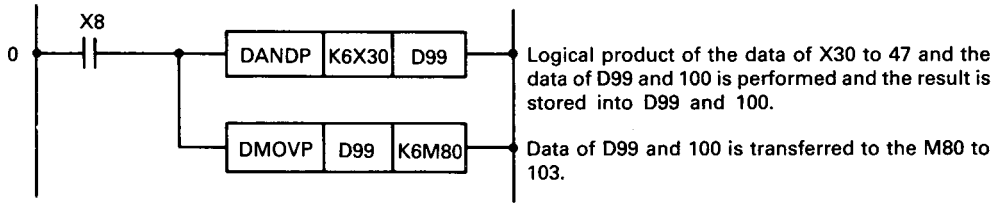
Step Number	Instruction	Device			
0	LD	XA			
1	WANDP	K3X10	D33	K3Y30	
8	END				



IB (NA) 66147-A

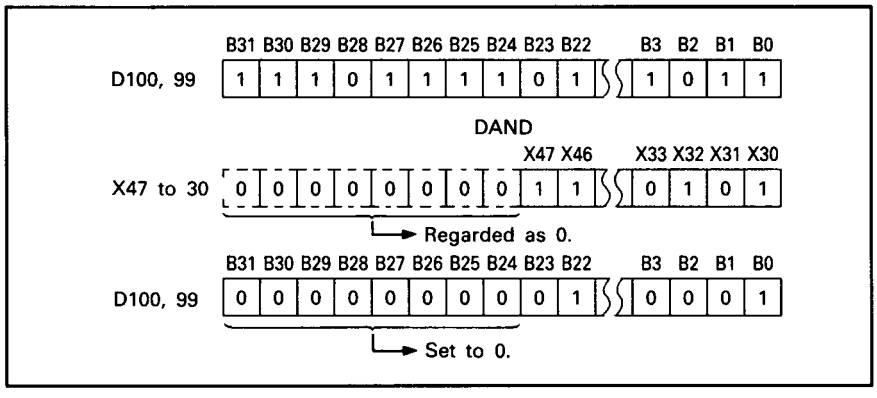
DAND

1) Program which performs logical product of the 24-bit data of X30 to 47 and the data of D99 and 100, then transfers the result to the M80 to 103 when X8 turns on.

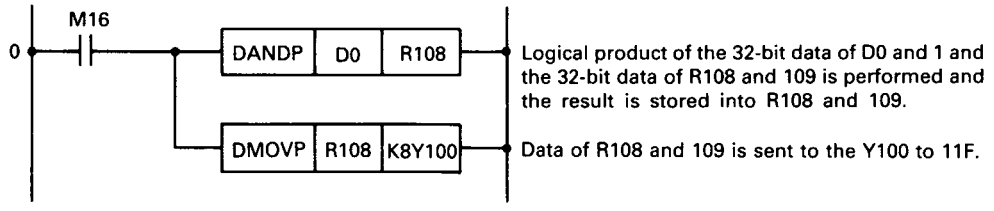


Coding

Step Number	Instruction	Device			
0	LD	X8			
1	DANDP	K6X30	D99		
10	DMOVP	D99	K6M80		
17	END				



2) Program which performs logical product of the 32-bit data of D0 and 1 and the 32-bit data of R108 and 109, and sends the result to the Y100 to 11F when M16 turns on.



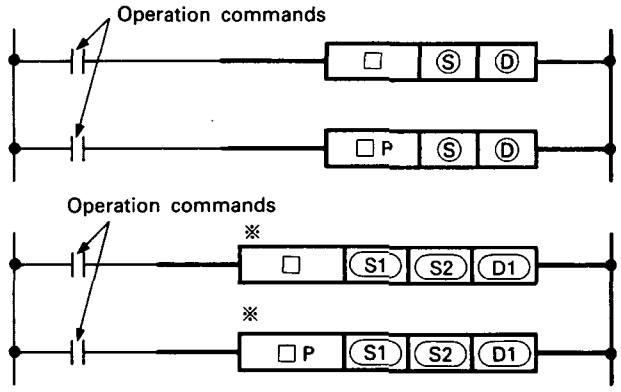
Coding

Step Number	Instruction	Device			
0	LD	M16			
1	DANDP	D0	R108		
10	DMOVP	R108	M8Y100		
17	END				

7.1.2 16-, 32-bit data logical add (WOR, WORP, DOR, DORP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
WOR	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5 to 7	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
	(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DOR	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1 to K8	9					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



□ Indicates the instruction symbol.

WOR, DOR

Setting data

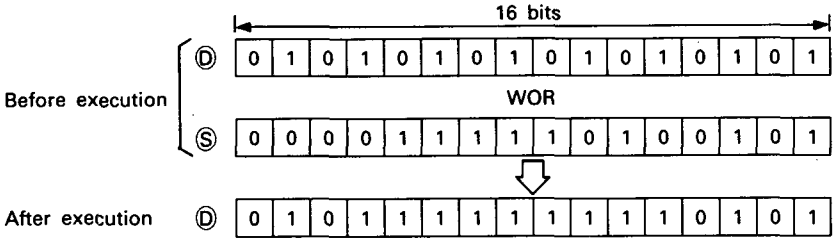
Ⓢ (S1) (S2) Ⓓ	Data for which logical add will be performed or head number of device which stores data
(D1)	Head number of device which will store the result of logical add

WOR may only be used in the areas marked ※.

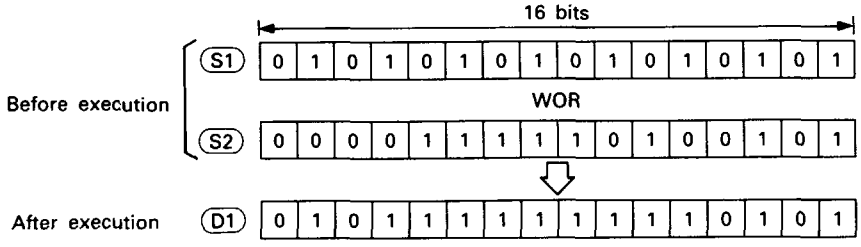
Functions

WOR

(1) Performs the logical add of the 16-bit data of device specified at Ⓓ and the 16-bit data of device specified at Ⓢ per bit, and stores the result into the device specified at Ⓓ.



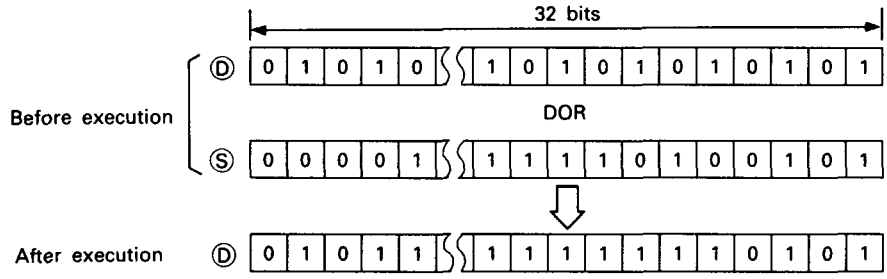
(2) Performs the logical add of the 16-bit data of device specified at (S1) and the 16-bit data of device specified at (S2) per bit, and stores the result into the device specified at (D1).



(3) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

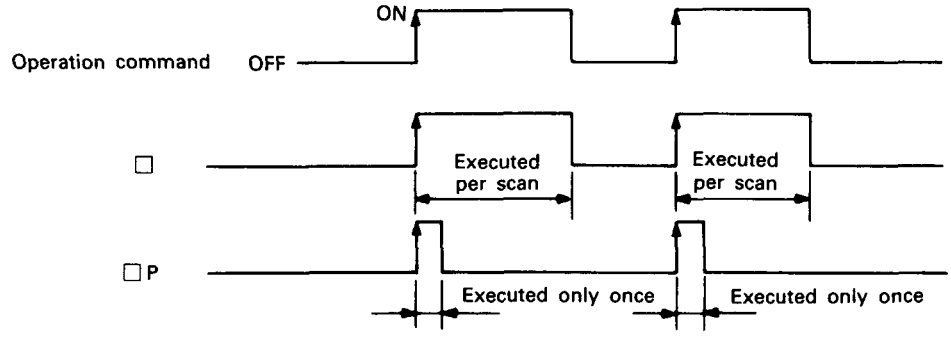
DOR

(1) Performs the logical add of the 32-bit data of device specified at Ⓓ and the 32-bit data of device specified at Ⓢ per bit, and stores the result into the device specified at Ⓓ.



(2) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

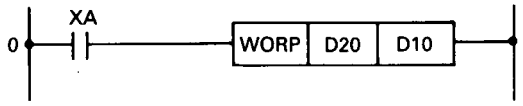
Execution Conditions



Program Examples

WOR

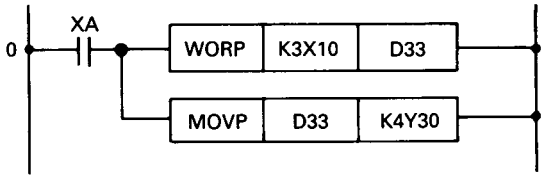
1) Program which performs logical add of the data of D10 and that of D20, and stores the result to D10 when XA turns on.



Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WORP	D20	D10		
6	END				

2) Program which performs logical add of the data of X10 to 1B and the data of D33, and sends the result to the Y30 to 3F when XA turns on.



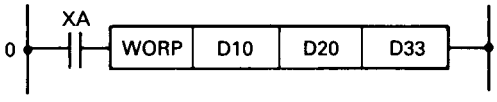
Logical add of the data of X10 to 1B and the data of D33 is performed and the result is stored into D33.

Data of D33 is sent to the Y30 to 3F.

Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WORP	K3X10	D33		
6	MOVP	D33	K4Y30		
11	END				

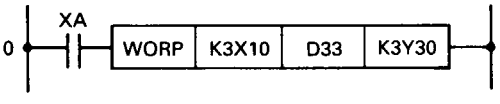
3) Program which performs logical add of the data of D10 and that of D20, and stores the result to D33 when XA turns on.



Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WORP	D10	D20	D33	
8	END				

4) Program which performs logical add of the data of X10 to 1B and the data of D33, and sends the result to the Y30 to 3B when XA turns on.

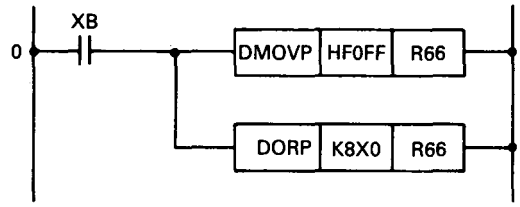


Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WORP	K3X10	D33	K3Y30	
8	END				

DOR

1) Program which performs logical add of the 32-bit data of X0 to 1F and the hexadecimal number of F0FF and stores the result to R66 and 67 when XB turns on.



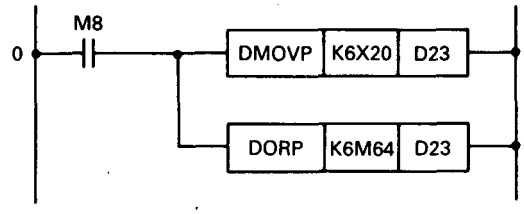
Hexadecimal number of F0FF is stored into R66 and 67.

Logical add of the 32-bit data of X0 to 1F and the 32-bit data of R66 and 67 is performed and the result is stored into R66 and 67.

Coding

Step Number	Instruction	Device			
0	LD	XB			
1	DMOVP	HF0FF	R66		
8	DORP	K8X0	R66		
17	END				

2) Program which performs logical add of the 24-bit data of M64 to 87 and the 24-bit data of X20 to 37 and stores the result to D23 and 24 when M8 turns on.



24-bit data of X20 to 37 is stored into D23 and 24.

Logical add of the 24-bit data of M64 to 87 and the 24-bit data of D23 and 24 is performed and the result is stored into D23 and 24.

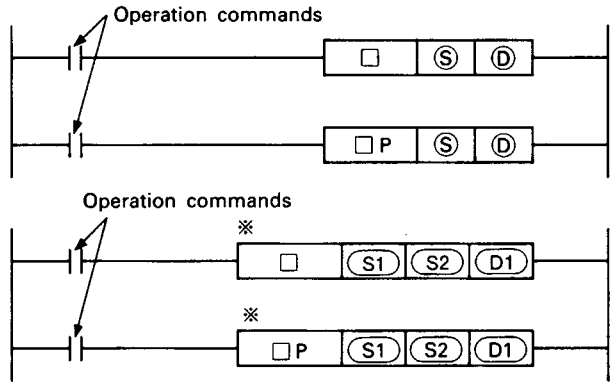
Coding

Step Number	Instruction	Device			
0	LD	M8			
1	DMOVP	K6Y20	D23		
8	DORP	K6M64	D23		
17	END				

7.1.3 16-, 32-bit data exclusive logical add (WXOR, WXORP, DXOR, DXORP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
WXOR	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5 to 7	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
	(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DXOR	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○		○	○				K1 to K8	9					
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○		○													



□ Indicates the instruction symbol.

WXOR, DXOR

Setting data

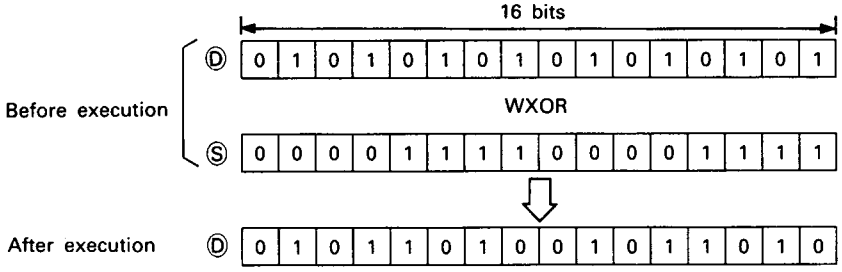
Ⓢ (S1) (S2) Ⓓ	Data for which exclusive OR will be performed or head number of device which stores data
(D1)	Head number of device which will store the result of exclusive OR

WXOR may only be used in the areas marked ※.

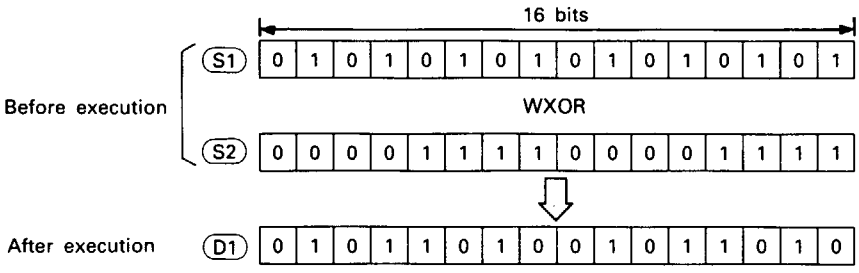
Functions

WXOR

(1) Performs the exclusive OR of the 16-bit data of device specified at Ⓓ and the 16-bit data of device specified at Ⓢ per bit, and stores the result into the device specified at Ⓓ.



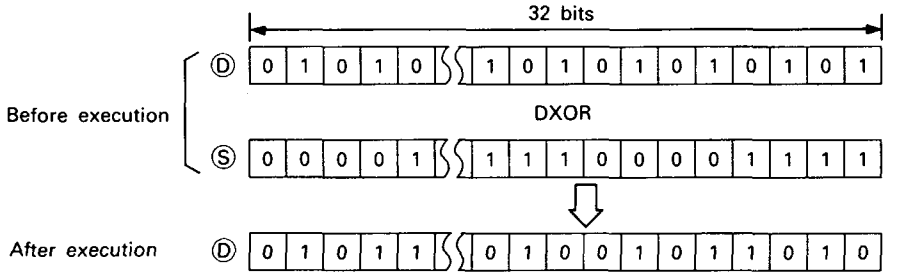
(2) Performs the exclusive OR of the 16-bit data of device specified at (S1) and the 16-bit data of device specified at (S2) per bit, and stores the result into the device specified at Ⓓ.



(3) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

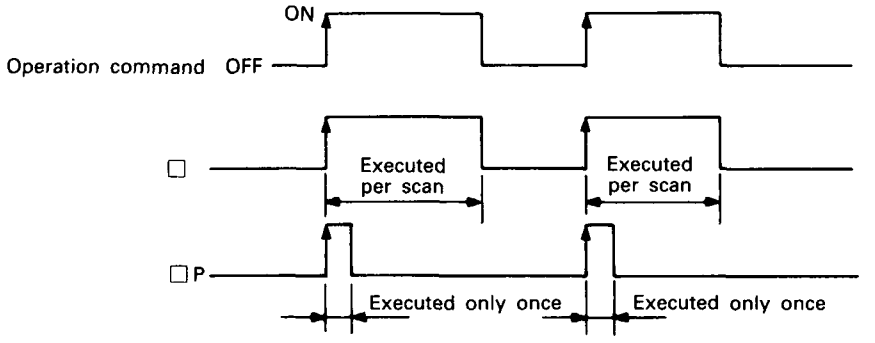
DXOR

(1) Performs the exclusive OR of the 32-bit data of device specified at Ⓓ and the 32-bit data of device specified at Ⓢ per bit, and stores the result into the device specified at Ⓓ.



(2) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

Execution Conditions



Program Examples

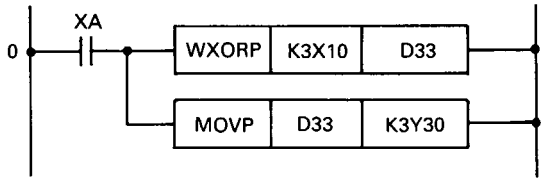
WXOR

1) Program which performs exclusive OR of the data of D10 and that of D20, and stores the result to D33 when XA turns on.

Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WXORP	D20	D10		
6	END				

2) Program which performs the exclusive OR of the data of X10 to 1B and the data of D33, and sends the result to the Y30 to 3B when XA turns on.



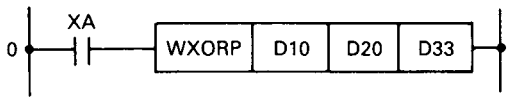
Exclusive OR of the data of X10 to 1B and the data of D33 is performed and the result is stored into D33.

Data of D33 is sent to Y30 to 3F.

Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WXORP	K3X10	D33		
6	MOVP	D33	K3Y30		
11	END				

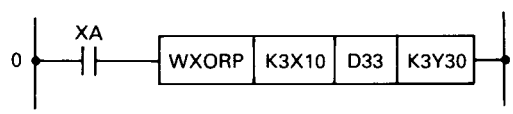
3) Program which performs exclusive OR of the data of D10 and that of D20, and stores the result to D33 when XA turns on.



Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WXORP	D10	D20	D33	
8	END				

4) Program which performs exclusive OR of the data of X10 to 1B and the data of D33, and sends the result to the Y30 to 3B when XA turns on.

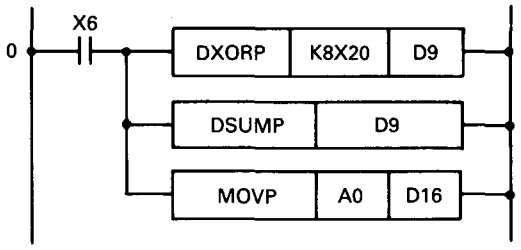


Coding

Step Number	Instruction	Device			
0	LD	XA			
1	WXORP	K3X10	D33	K3Y30	
8	END				

DXOR

1) Program which compares the 32-bit data of X20 to 3F and the bit pattern of data of D9 and 10, and stores the number of different bits to D16 when X7 turns on.



Exclusive OR of the 32-bit data of X20 to 3F and the data of D9 and 10 is performed.

Comparing 32-bit data of D9 and 10, the total number of "1" bits is stored into A0.

Data of A0 is stored into D16.

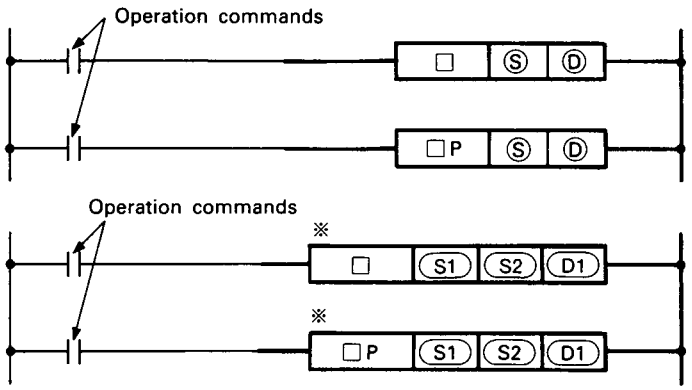
Coding

Step Number	Instruction	Device			
0	LD	X6			
1	DXORP	K8X20	D9		
10	DSUMP	D9			
13	MOVP	A0	D16		
18	END				

7.1.4 16-, 32-bit data NOT exclusive logical add (WXNR, WXNRP, DXNR, DXNRP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
WXNR	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	5 to 7	○	○		○	○
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
	(S1)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(S2)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○										
	(D1)		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												
DXNR	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○			○		○	○									
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○			○						K1 to K8	9					



□ Indicates the instruction symbol.

WXNR, DXNR

Setting data

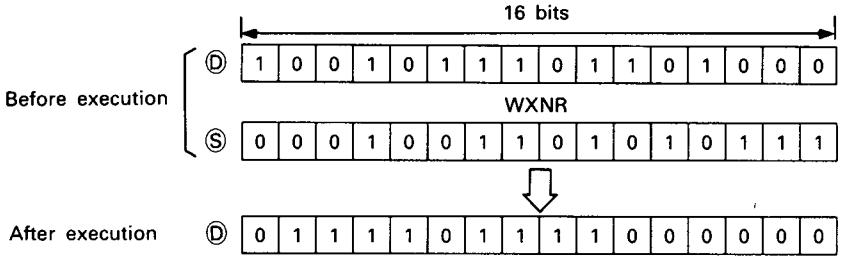
Ⓢ (S1) (S2) Ⓓ	Data for which exclusive NOR will be performed or head number of device which stores data
(D1)	Head number of device which will store the result of exclusive NOR

WXNR may only be used in the areas marked ※.

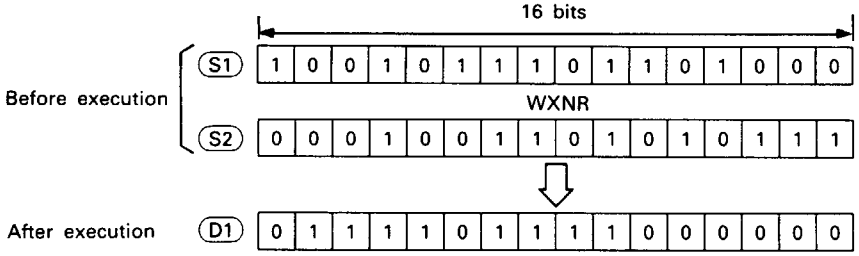
Functions

WXNR

(1) Performs the exclusive NOR of the 16-bit data of device specified at Ⓓ and the 16-bit data of device specified at Ⓢ and stores the result into the device specified at Ⓓ.



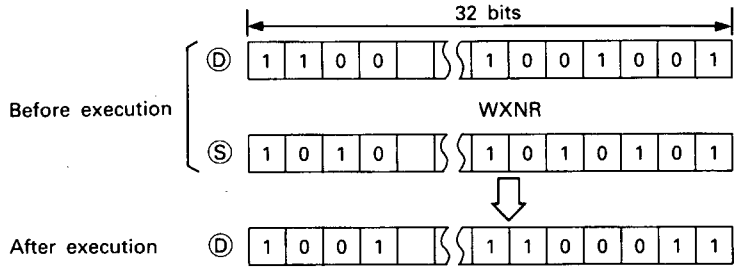
(2) Performs the exclusive NOR of the 16-bit data of device specified at (S1) and the 16-bit data of device specified at (S2) and stores the result into the device specified at Ⓓ.



(3) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

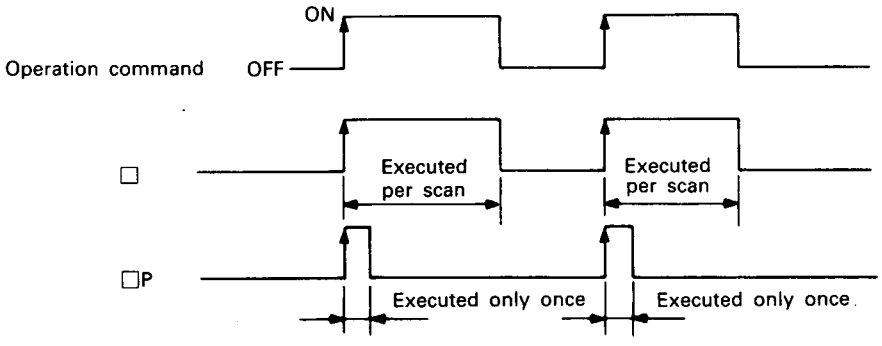
DXNR

(1) Performs the exclusive NOR of the 32-bit data of device specified at Ⓓ and the 32-bit data of device specified at Ⓢ and stores the result into the device specified at Ⓓ.



(2) When operation is performed, the digits of bit device higher than the specified are regarded as 0.

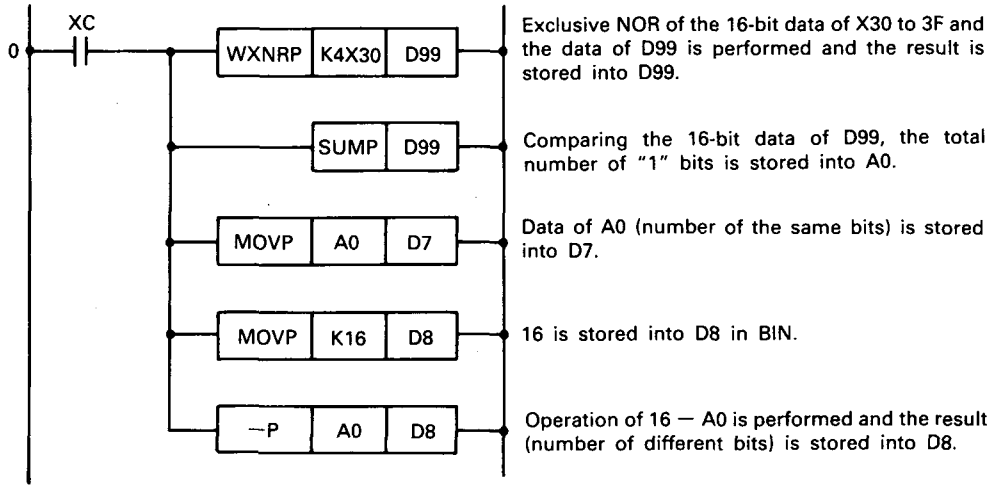
Execution Conditions



Program Examples

WXNR

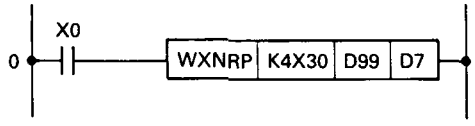
1) Program which compares the bit pattern of the 16-bit data of X30 to 3F and that of the 16-bit data of D99 and stores the number of the same bit patterns and the number of different bit patterns to D7 and 8, respectively, when XC turns on.



Coding

Step Number	Instruction	Device			
0	LD	XC			
1	WXNRP	K4X30	D99		
6	SUMP	D99			
9	MOVP	A0	D7		
14	MOVP	K16	D8		
19	—P	A0	D8		
24	END				

2) Program which compares the bit pattern of the 16-bit data of X30 to 3F and that of the data of D99 and stores the result to D7 when X0 turns on.

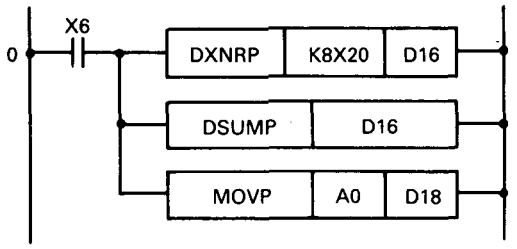


Coding

Step Number	Instruction	Device			
0	LD	X0			
1	WXNRP	K4X30	D99	D7	
8	END				

DXNR

1) Program which compares the bit pattern of the 32-bit data of X20 to 3F and that of the data of D16 and 17, and stores the number of the same bit patterns to D18 when X6 turns on.



Exclusive NOR of the 32-bit data of X20 to 3F and the data of D16 and 17 is performed.

Among 32-bit data of D16 and 17, the total number of "1" bits is stored into A0.

Data of A0 is stored into D18.

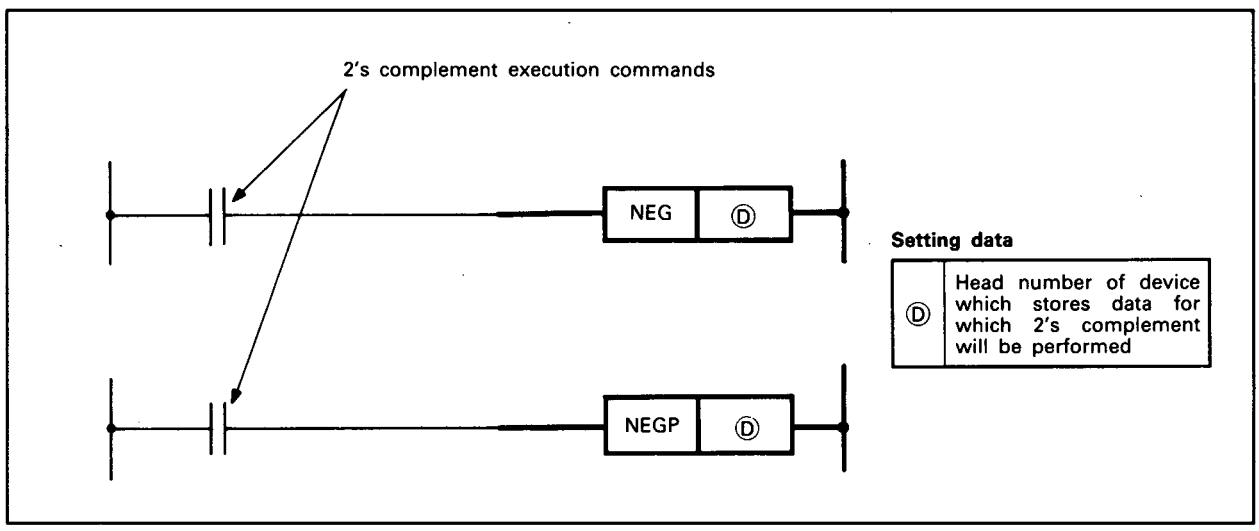
Coding

Step Number	Instruction	Device			
0	LD	X6			
1	DXNRP	K8X20	D16		
10	DSUMP	D16			
13	MOVP	A0	D18		
18	END				

7.1.5 BIN 16-bit data 2's complement (NEG, NEGP)

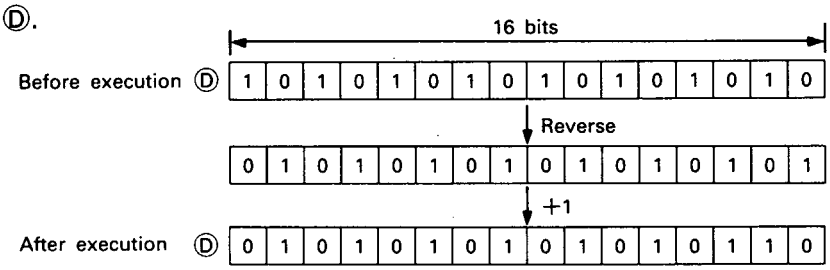
Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4	3		○		○	○



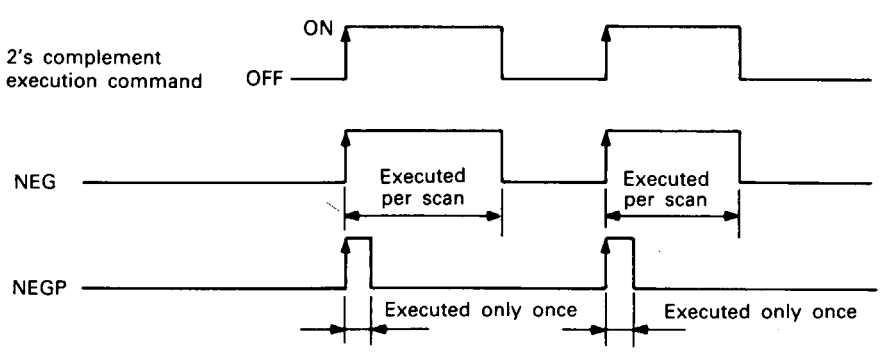
Functions

(1) Reverses the 16-bit data of device specified at Ⓓ, adds 1 to the value, and stores the addition result to the device specified at Ⓓ.



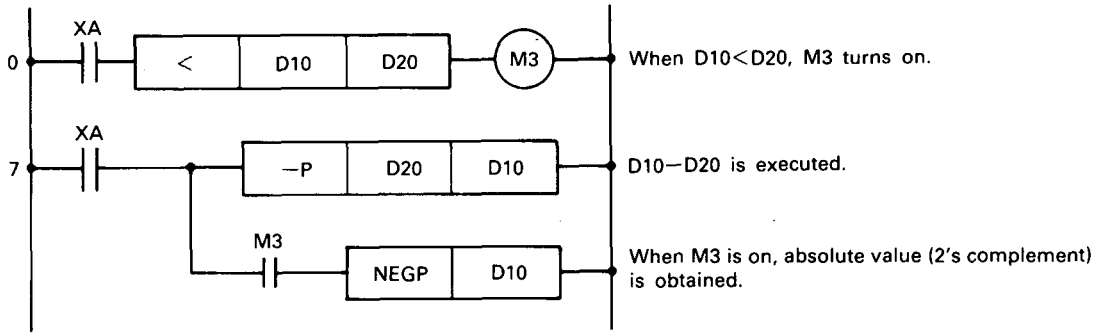
(2) Used to obtain the absolute value of a negative BIN value.

Execution Conditions



Program Example

1) Program which calculates "D10 — D20" when XA turns on, and obtains the absolute value when the result is negative.



Coding

Step Number	Instruction	Device			
0	LD	XA			
1	AND<	D10	D20		
6	OUT	M3			
7	LD	XA			
8	—P	D20	D10		
13	AND	M3			
14	NEGP	D10			
17	END				

MEMO

7.2 Rotation Instructions

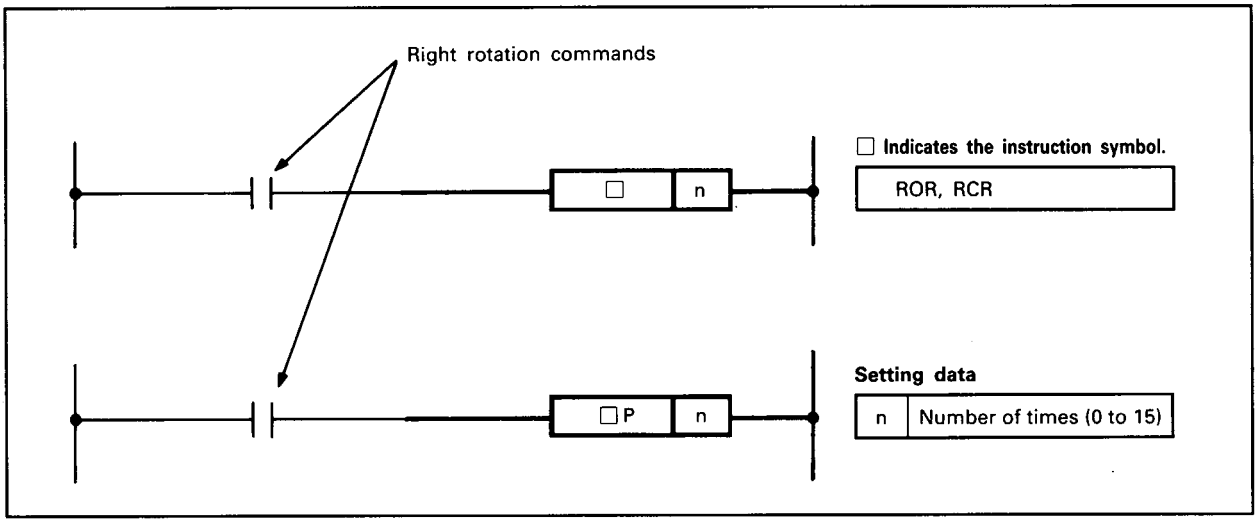
The rotation instructions rotate the data stored in the accumulator.

Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page
Right rotation	ROR	7-22 to 7-23	Left rotation	ROL	7-24 to 7-25
	RORP	7-22 to 7-23		ROLP	7-24 to 7-25
	RCR	7-22 to 7-23		RCL	7-24 to 7-25
	RCRP	7-22 to 7-23		RCLP	7-24 to 7-25
	DROR	7-26 to 7-27		DROL	7-28 to 7-29
	DRORP	7-26 to 7-27		DROLP	7-28 to 7-29
	DRCR	7-26 to 7-27		DRCL	7-28 to 7-29
	DRCRP	7-26 to 7-27		DRCLP	7-28 to 7-29

7.2.1 16-bit data right rotation (ROR, RORP, RCR, RCRP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n																	○	○					3		○	○		

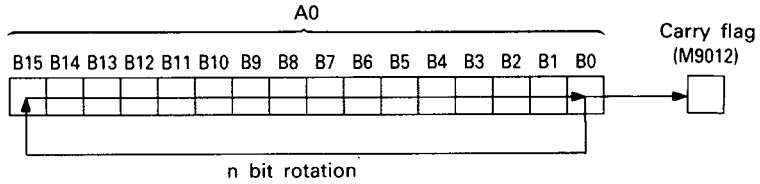


Functions

ROR

Rotates the data of A0 "n" bits to the right, without including the carry flag.

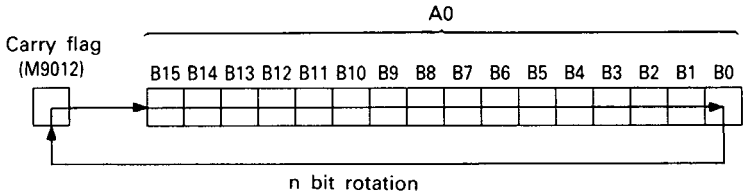
○ The carry flag is 1 or 0 depending on the status prior to the execution of ROR.



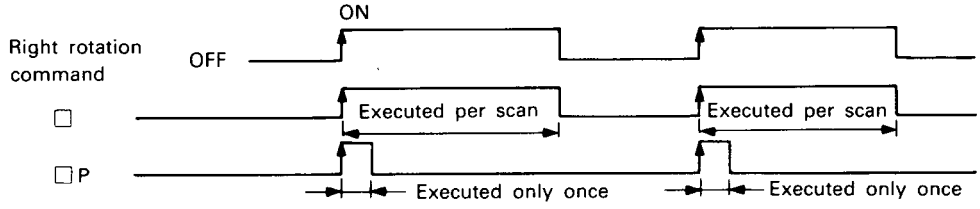
RCR

Rotates the data of A0 "0" bits to the right, including the carry flag.

○ The carry flag is 1 or 0 depending on the status prior to the execution of RCR.



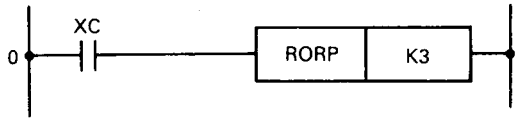
Execution Conditions



Program Examples

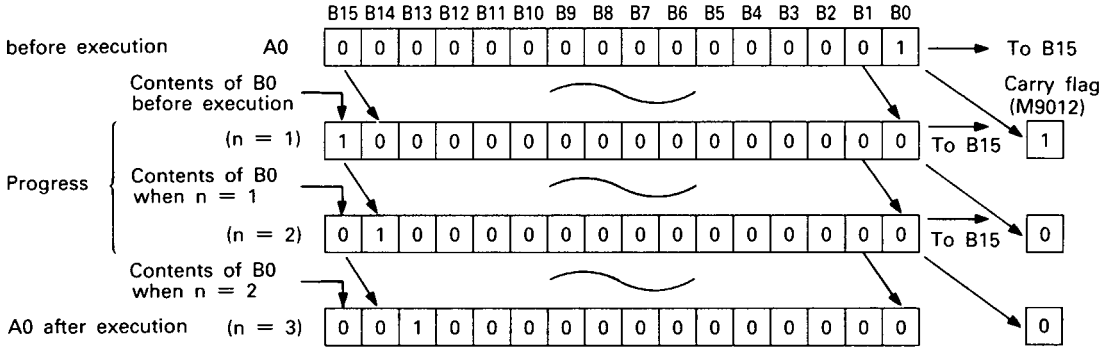
ROR

Program which rotates the contents of A0 three bits to the right when XC turns on.



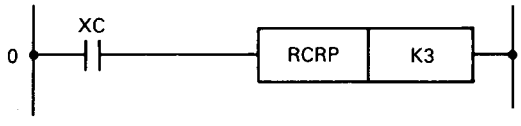
Coding

Step Number	Instruction	Device			
0	LD	XC			
1	RORP	K3			
4	END				



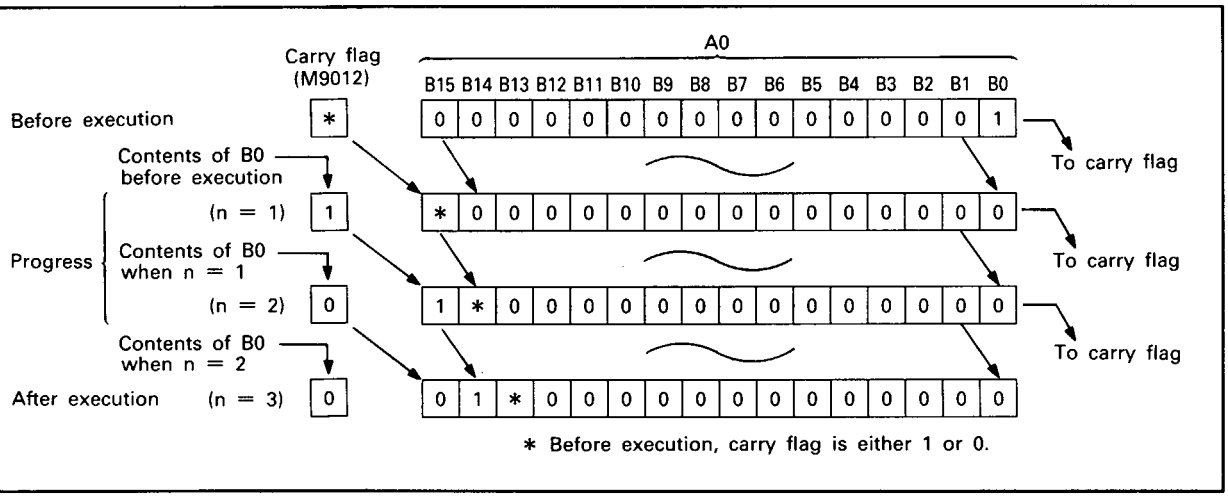
RCR

Program which rotates the contents of A0 three bits to the right when XC turns on.



Coding

Step Number	Instruction	Device			
0	LD	XC			
1	RCRP	K3			
4	END				

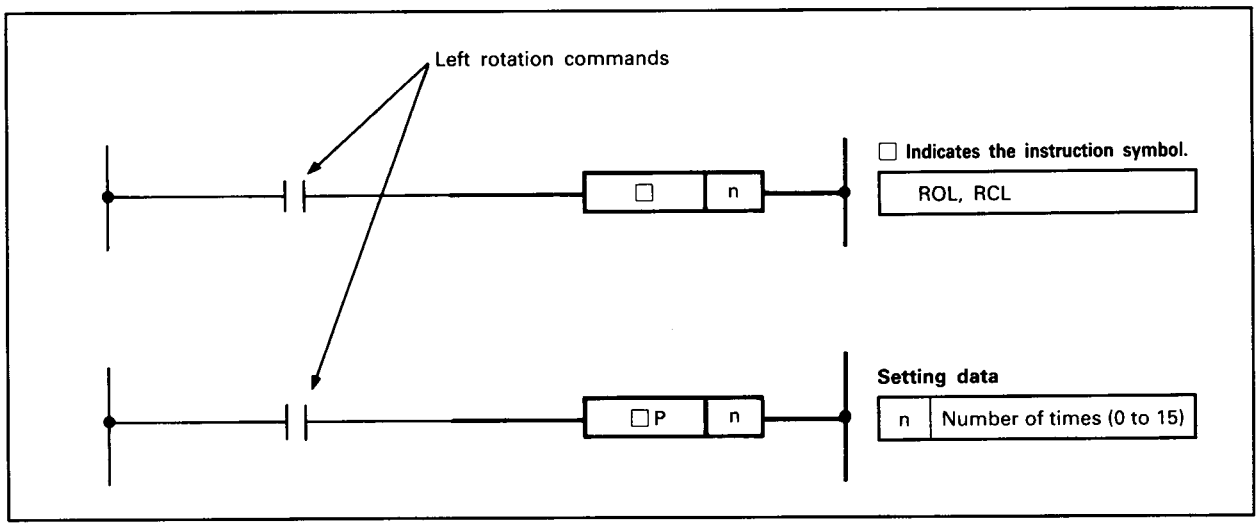


IB (NA) 66147-A

7.2.2 16-bit data left rotation (ROL, ROLP, RCL, RCLP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n																	○	○					3		○	○		

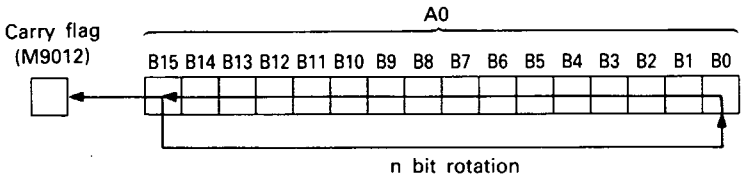


Functions

ROL

Rotates the data of A0 "n" bits to the left, without including the carry flag.

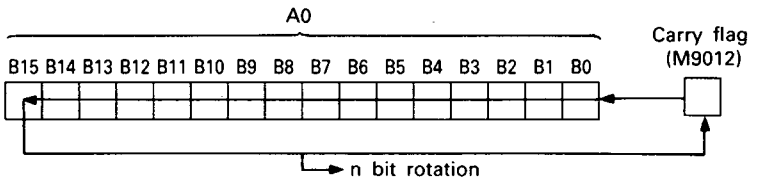
○ The carry flag is 1 or 0 depending on the status prior to the execution of ROL.



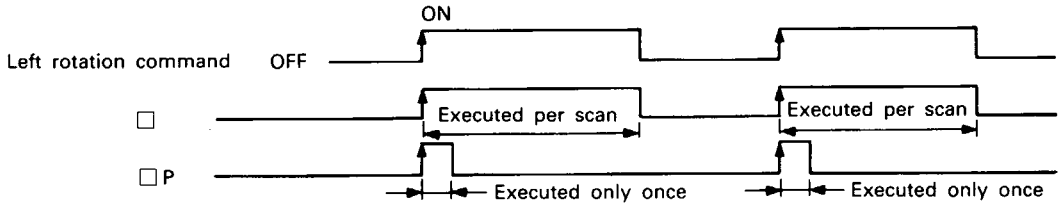
RCL

Rotates the data of A0 "0" bits to the left, including the carry flag.

○ The carry flag is 1 or 0 depending on the status prior to the execution of RCL.



Execution Conditions



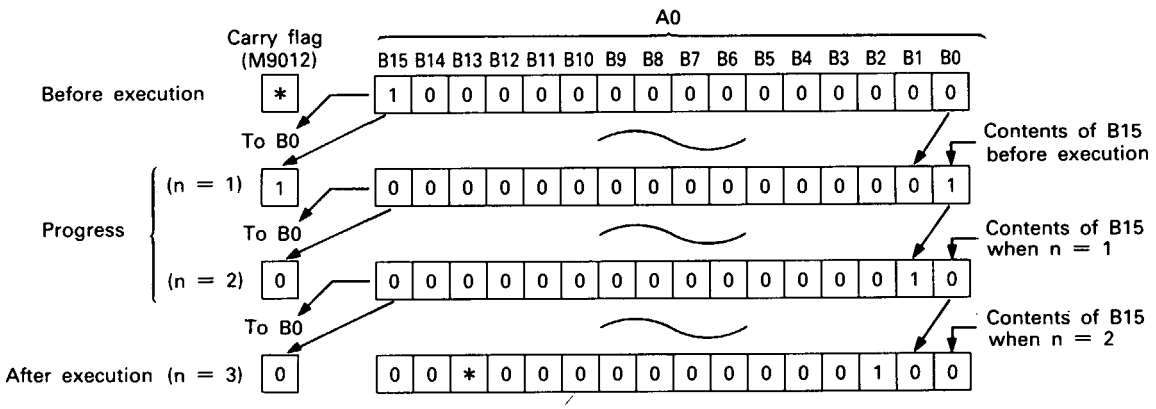
Program Examples

ROL

Program which rotates the contents of A0 three bits to the left when XC turns on.

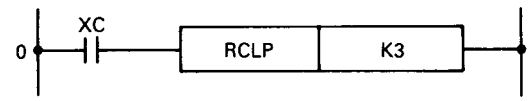
Coding

Step Number	Instruction	Device			
0	LD	XC			
1	ROLP	K3			
4	END				



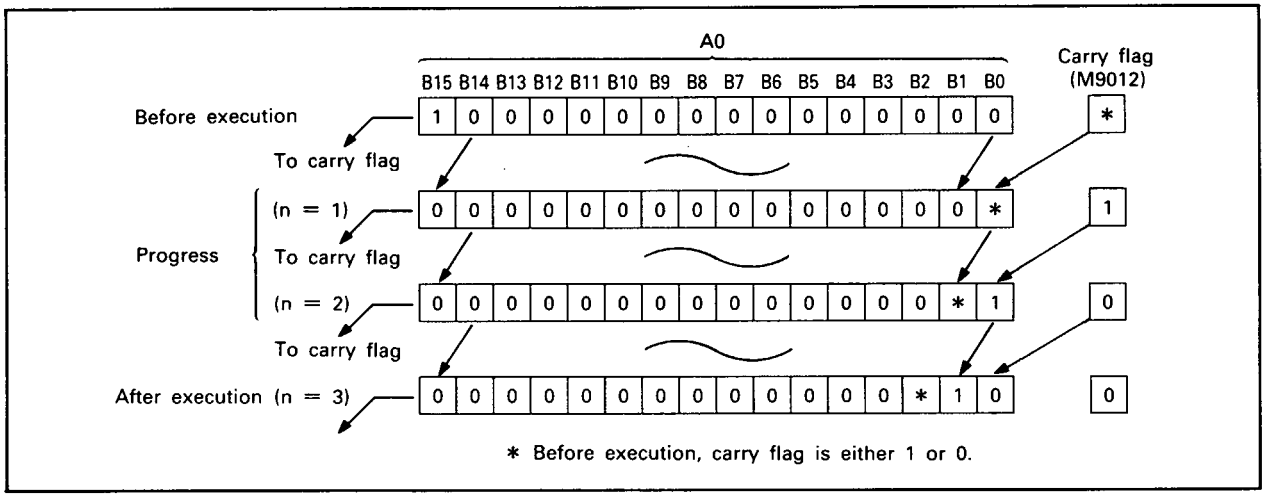
RCL

Program which rotates the contents of A0 three bits to the left when XC turns on.



Coding

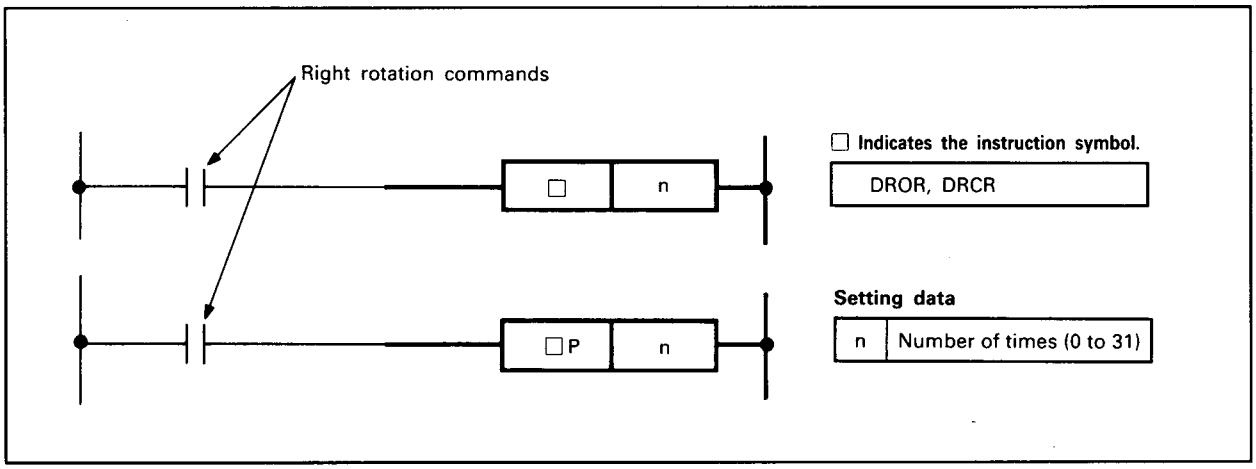
Step Number	Instruction	Device			
0	LD	XC			
1	RCLP	K3			
4	END				



7.2.3 32-bit data right rotation (DROR, DRORP, DRCR, DRCRP)

Processing Unit	Applicable CPU			
32 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n																	○	○					3		○	○		

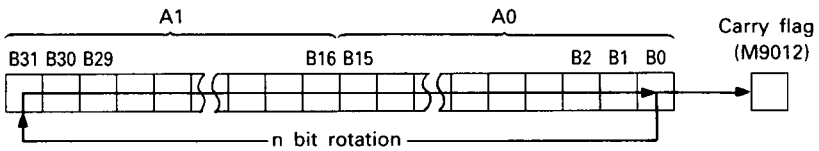


Functions

DROR

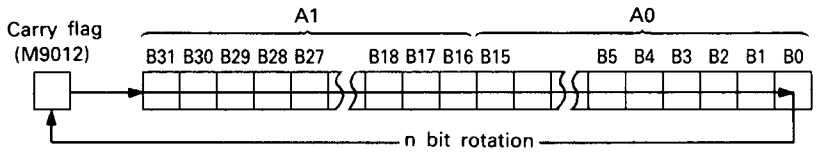
Rotates the data of A0 and 1 "n" bits to the right, without including the carry flag.

○ The carry flag is 1 or 0 depending on the status prior to the execution of DROR.



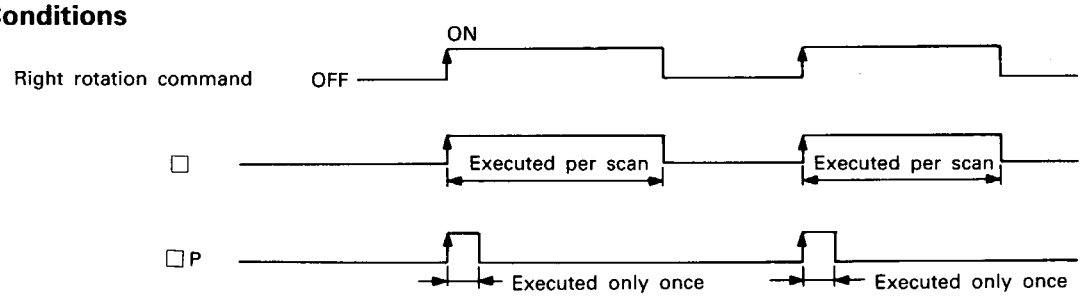
DRCR

Rotates the data of A0 and 1 "0" bits to the right, including the carry flag.



○ The carry flag is 1 or 0 depending on the status prior to the execution of DRCR.

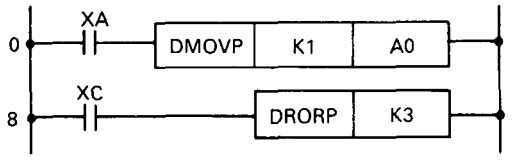
Execution Conditions



Program Examples

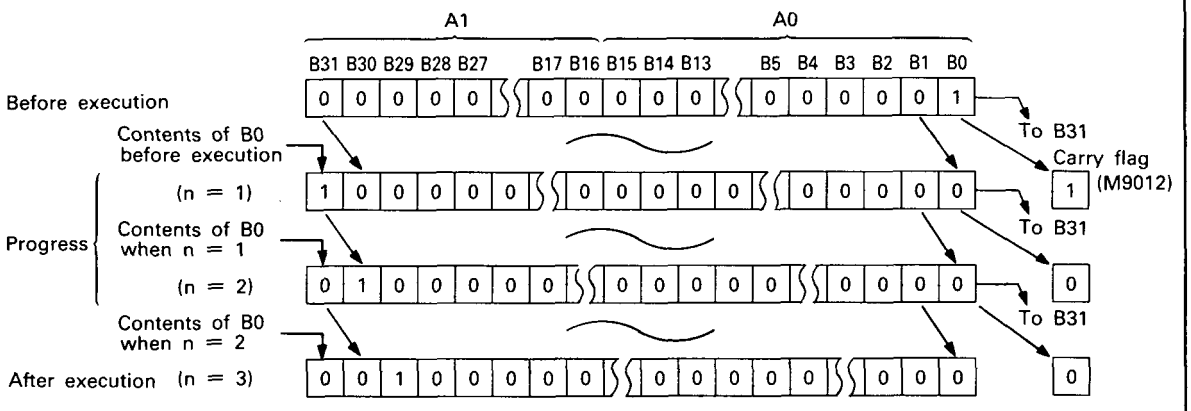
DROR

Program which rotates the contents of A0 and 1 three bits to the right when XC turns on.



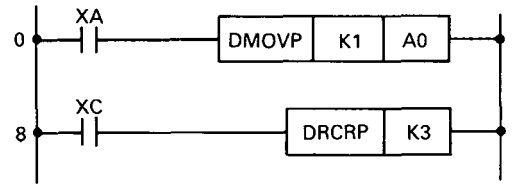
Coding

Step Number	Instruction	Device			
0	LD	XA			
1	DMOVP	K1	A0		
8	LD	XC			
9	DRORP	K3			
12	END				



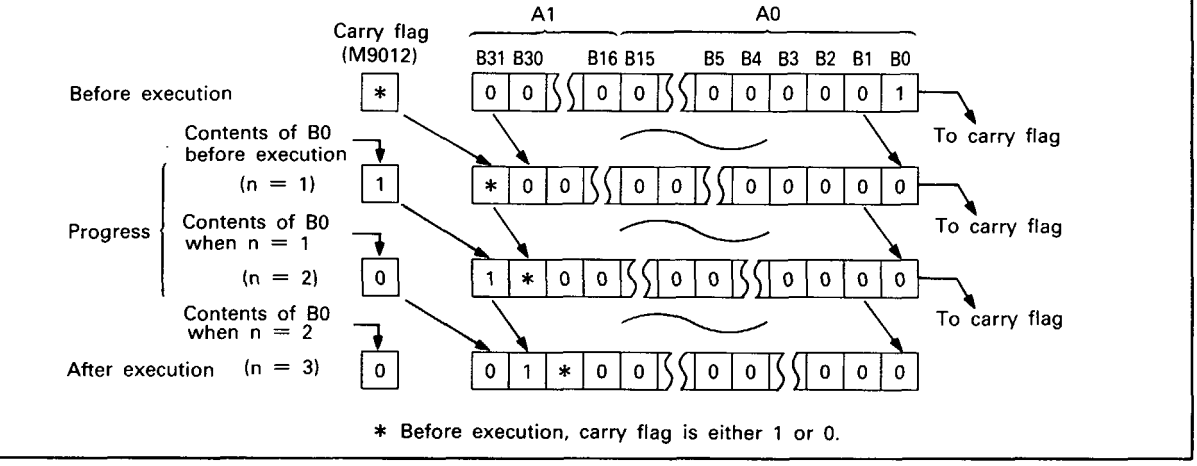
DRCR

Program which rotates the contents of A0 and 1 three bits to the right when XC turns on.



Coding

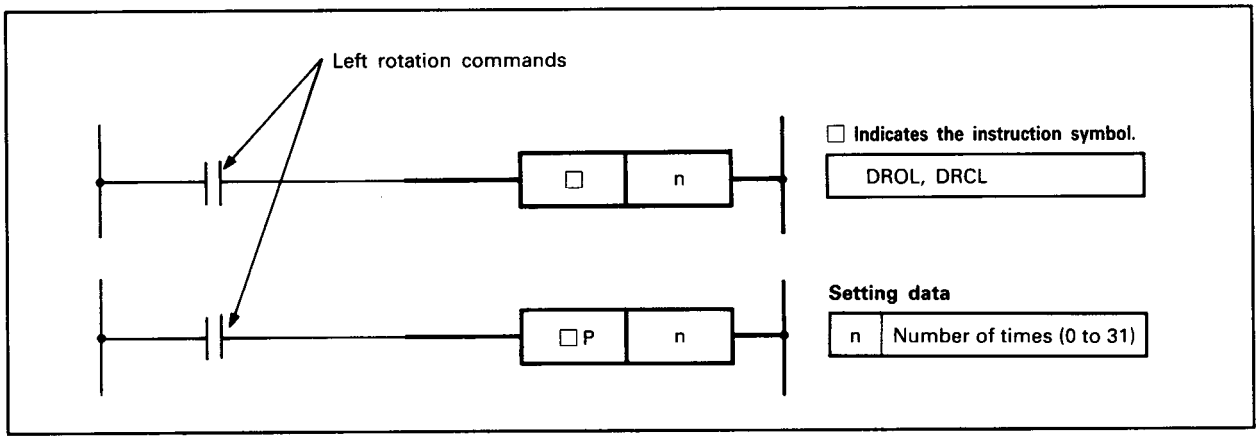
Step Number	Instruction	Device			
0	LD	XA			
1	DMOVP	K1	A0		
8	LD	XC			
9	DRCRP	K3			
12	END				



7.2.4 32-bit data left rotation (DROL, DROLP, DRCL, DRCLP)

Processing Unit	Applicable CPU			
32 bits	A1N	A2N	A3N	A3H

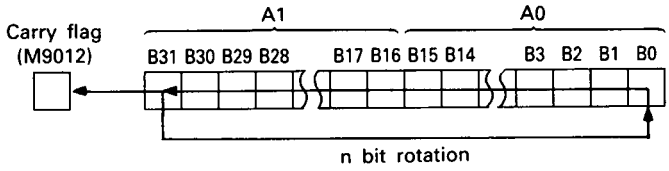
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n																	○	○					3		○	○		



Functions

DROL

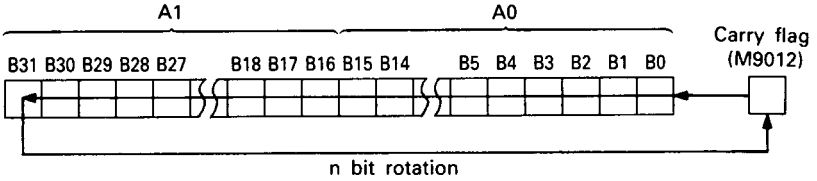
Rotates the data of A0 and 1 "n" bits to the left, without including the carry flag.



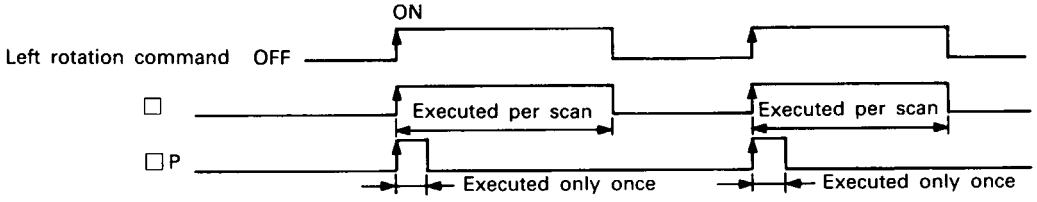
DRCL

Rotates the data of A0 and 1 "n" bits to the left, including the carry flag.

○ The carry flag is 1 or 0 depending on the status prior to the execution of DRCL.



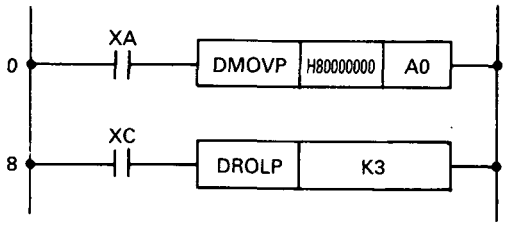
Execution Conditions



Program Examples

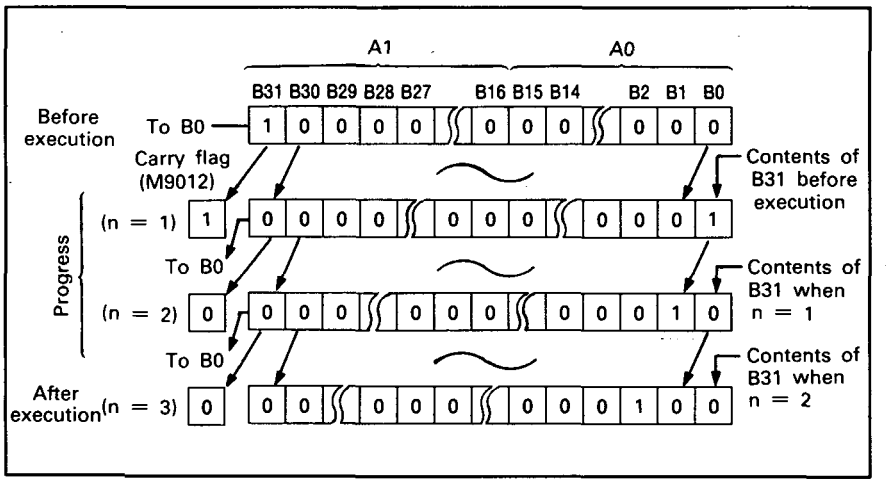
DROL

Program which rotates the contents of A0 and 1 three bits to the left when XC turns on.



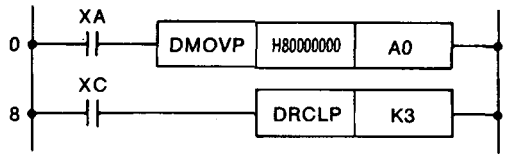
Coding

Step Number	Instruction	Device			
0	LD	XA			
1	DMOVP	H80000000	A0		
8	LD	XC			
9	DROLP	K3			
12	END				



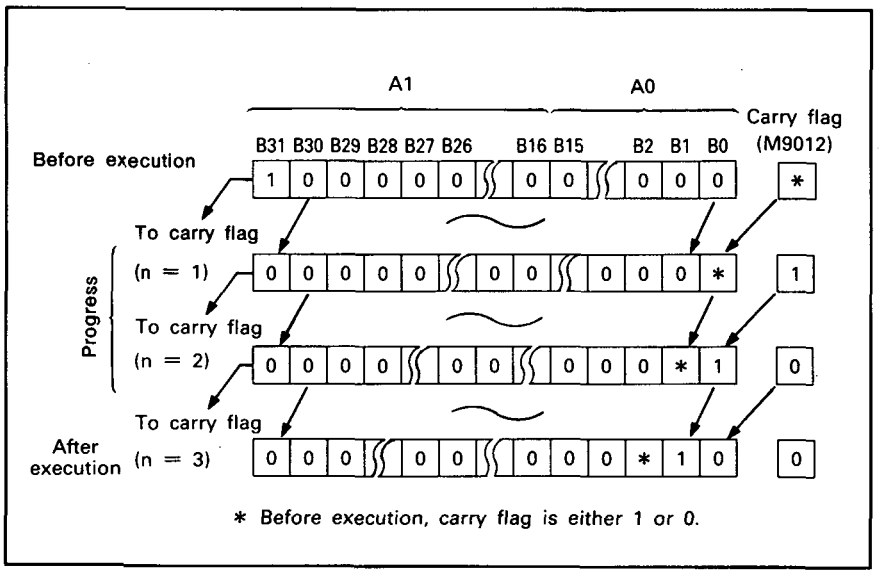
DRCL

Program which rotates the contents of A0 and 1 three bits to the left when XC turns on.



Coding

Step Number	Instruction	Device			
0	LD	XA			
1	DMOVP	H80000000	A0		
8	LD	XC			
9	DRCLP	K3			
12	END				



MEMO

7.3 Shift Instructions

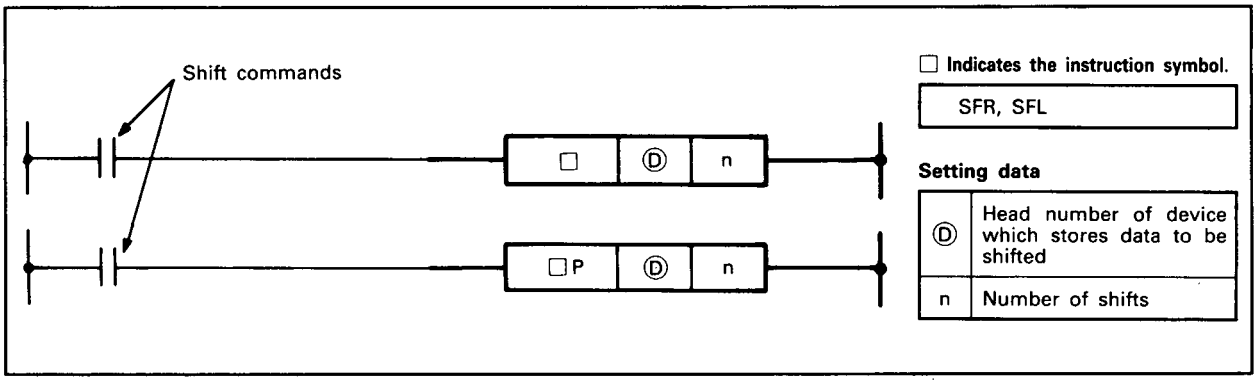
The shift instructions perform the shifting of data.

Classification	Instruction Symbol	Ref. Page	Classification	Instruction Symbol	Ref. Page
Right shift	SFR	7-31 to 7-32	Left shift	SFL	7-31 to 7-32
	SFRP	7-31 to 7-32		SFLP	7-31 to 7-32
	BSFR	7-33 to 7-34		BSFL	7-33 to 7-34
	BSFRP	7-33 to 7-34		BSFLP	7-33 to 7-34
	DSFR	7-35 to 7-36		DSFL	7-35 to 7-36
	DSFRP	7-35 to 7-36		DSFLP	7-35 to 7-36

7.3.1 16-bit data n-bit right shift, left shift (SFR, SFRP, SFL, SFLP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

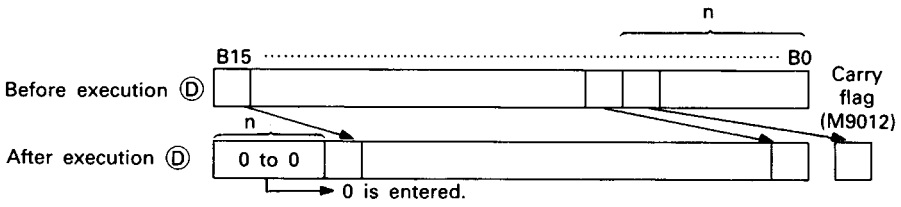
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4	5	○	○	○	○	○
n																	○	○										



Functions

SFR

(1) Shifts the 16-bit data of device specified at Ⓓ to the right by "n" bits.



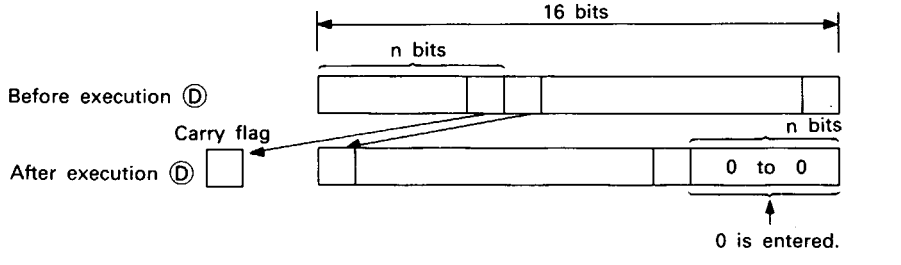
(2) "n" bits, which begin with the highest bit, change to 0.

(3) For T/C, the present value (count value) is shifted. (The shift of set value cannot be performed.)

SFL

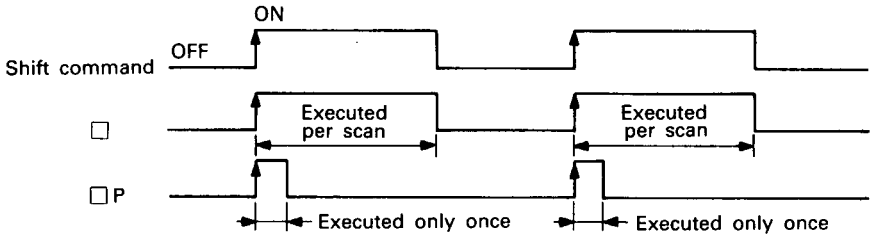
(1) Shifts the 16-bit data of device specified at Ⓓ to the left by "n" bits.

(2) "n" bits, which begin with the lowest bit, change to 0.



(3) In regards to T/C, the present value (count value) is shifted. (The shift of set value cannot be performed.)

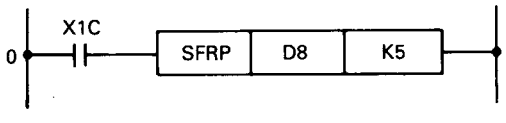
Execution Conditions



Program Examples

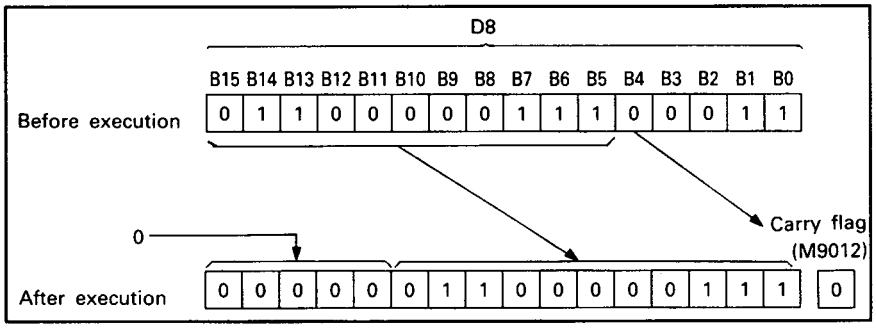
SFR

Program which shifts the contents of D8 five bits to the right when X1C turns on.



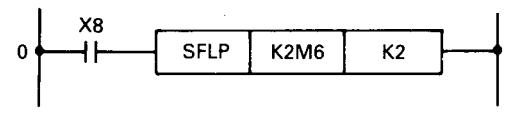
Coding

Step Number	Instruction	Device			
0	LD	X1C			
1	SFRP	D8	K5		
6	END				



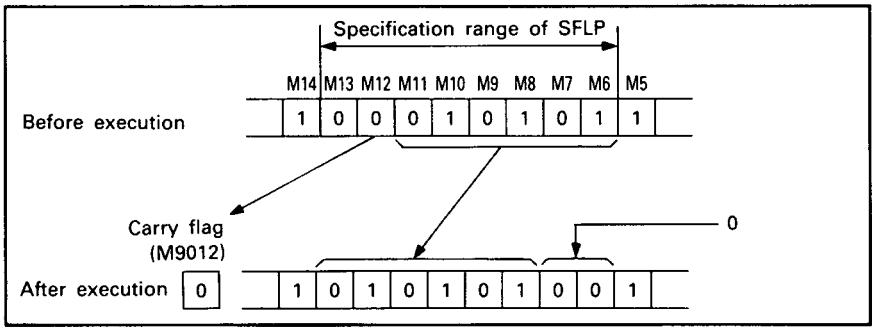
SFL

Program which shifts the data of M6 to 13 two bits to the left when X8 turns on.



Coding

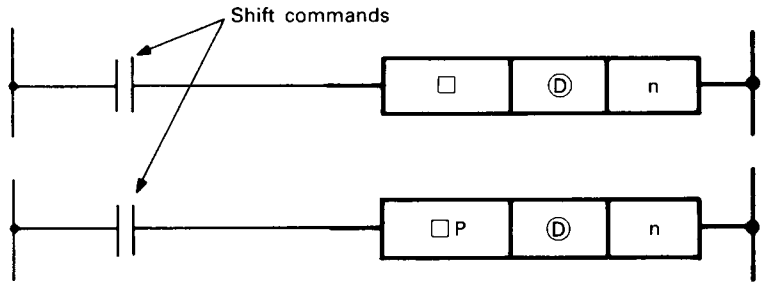
Step Number	Instruction	Device			
0	LD	X8			
1	SFLP	K2M6	K2		
6	END				



7.3.2 n-bit data 1-bit right shift, left shift (BSFR, BSFRP, BSFL, BSFLP)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ		○	○	○	○	○	○							-														
n																	○	○					7		○	○	○	○



□ Indicates the instruction symbol.

BSFR, BSFL

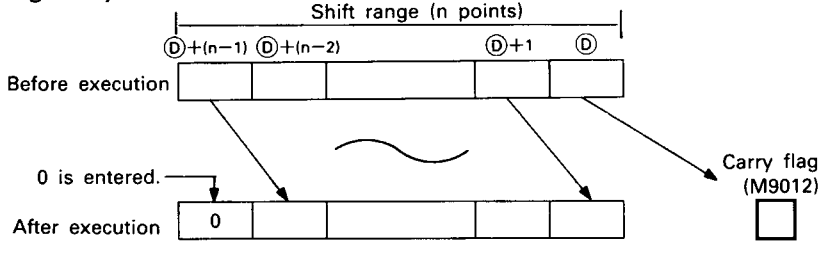
Setting data

Ⓓ	Head number of device which stores data to be shifted
n	Number of shifts

Functions

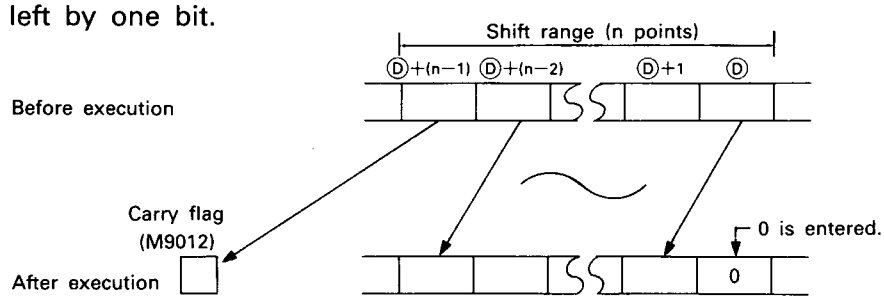
BSFR

Shifts "n" bits, which begins with the device specified at Ⓓ, to the right by one bit.

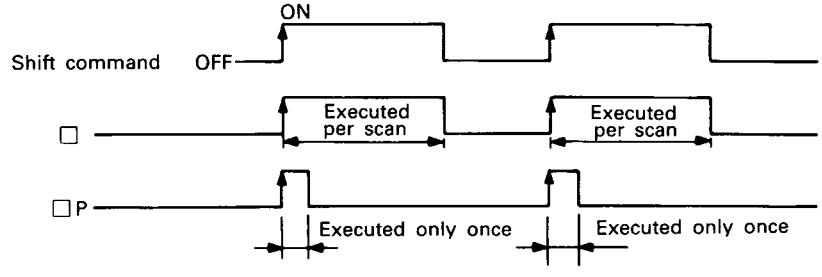


BSFL

Shifts "n" bits, which begin with the device specified at Ⓓ, to the left by one bit.



Execution Conditions



Operation Error

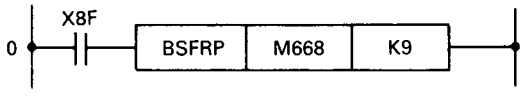
In the following case, operation error occurs and the error flag turns on.

○ "n" is a negative value.

Program Examples

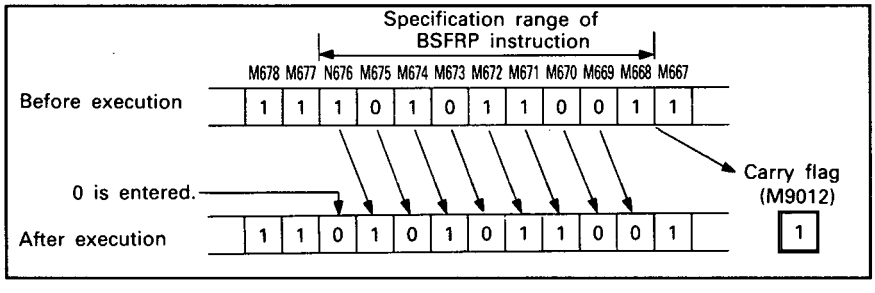
BSFR

Program which shifts the data of M668 to 676 to the right when X8F turns on.



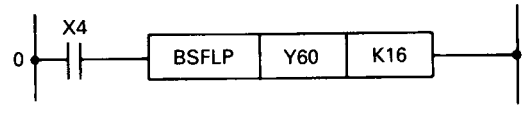
Coding

Step Number	Instruction	Device			
0	LD	X8F			
1	BSFRP	M668	K9		
8	END				



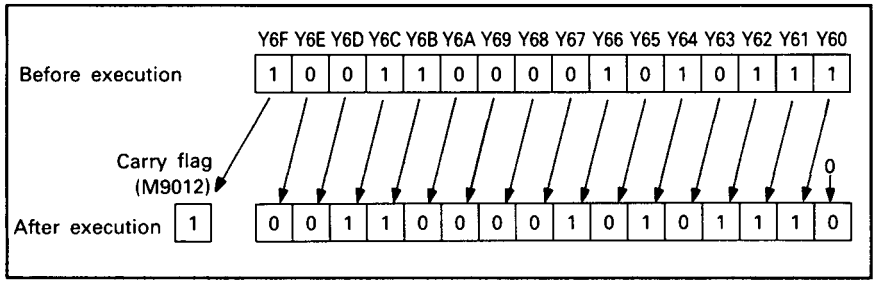
BSFL

Program which shifts the outputs of Y60 to 6F to the left when X4 turns on.



Coding

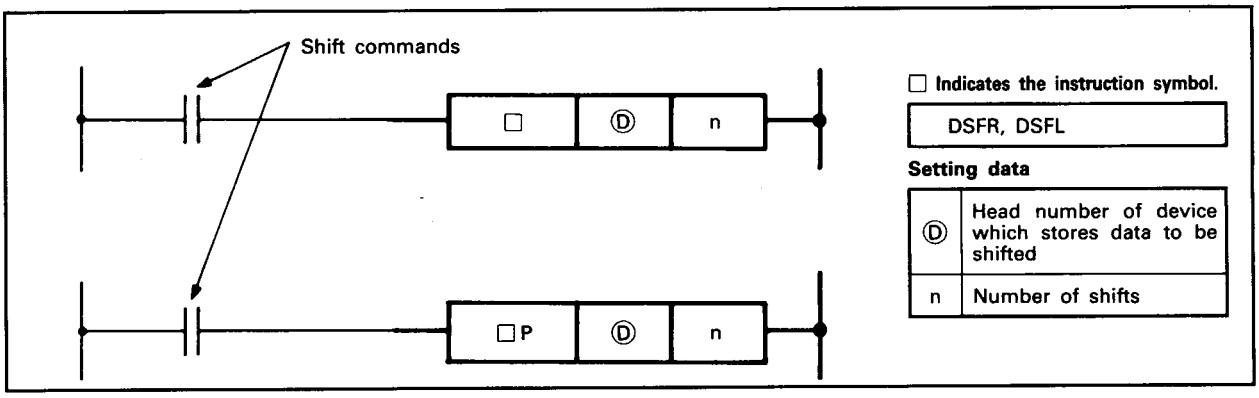
Step Number	Instruction	Device			
0	LD	X4			
1	BSFLP	Y60	K16		
8					



7.3.3 n-word data 1-word right shift, left shift (DSFR, DSFRP, DSFL, DSFLP)

Processing Unit	Applicable CPU			
1 word	A1N	A2N	A3N	A3H

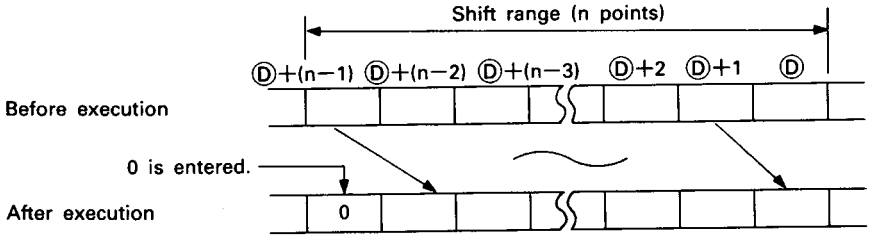
	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ								○	○	○	○	○										7			○		○	○
n																	○	○										



Functions

DSFR

(1) Shifts the word devices of "n" points, which begin with the device specified at Ⓓ, to the right by one bit.

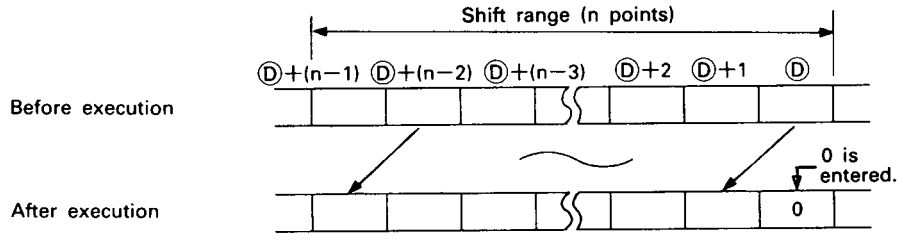


(2) The highest bit changes to 0.

(3) For T/C, the present value (count value) is shifted. (The shift of set value cannot be performed.)

DSFL

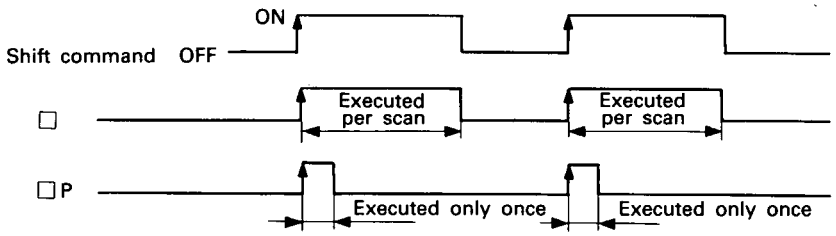
(1) Shifts the word devices of "n" points, which begin with the device specified at Ⓓ, to the left by one bit.



(2) The lowest bit changes to 0.

(3) In regards to T/C, the present value (count value) is shifted. (The shift of set value cannot be performed.)

Execution Conditions



Operation Error

In the following case, operation error occurs and the error flag turns on.

○ "n" is a negative value.

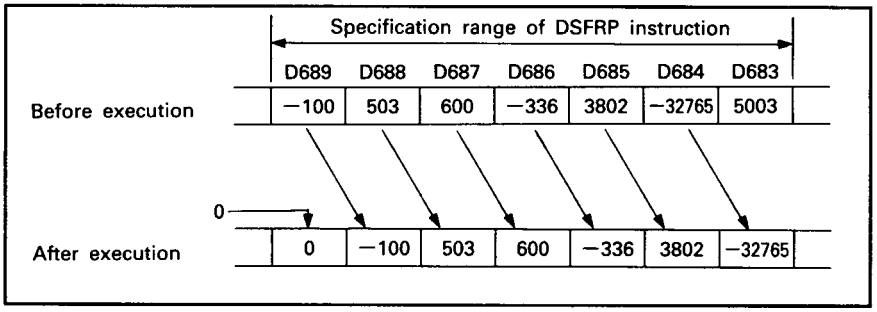
Program Examples

DSFR

Program which shifts the contents of D683 to 689 to the right when XB turns on.

Coding

Step Number	Instruction	Device			
0	LD	XB			
1	DSFRP	D683	K7		
8	END				

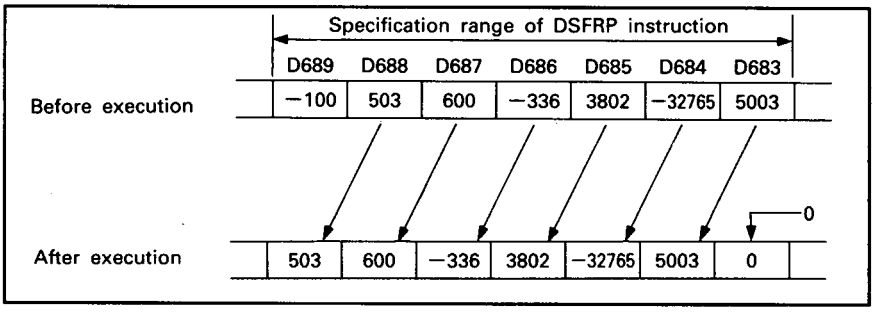


DSFL

Program which shifts the contents of D683 to 689 to the left when XB turns on.

Coding

Step Number	Instruction	Device			
0	LD	XB			
1	DSFLP	D683	K7		
8	END				



MEMO

7.4 Data Processing Instructions

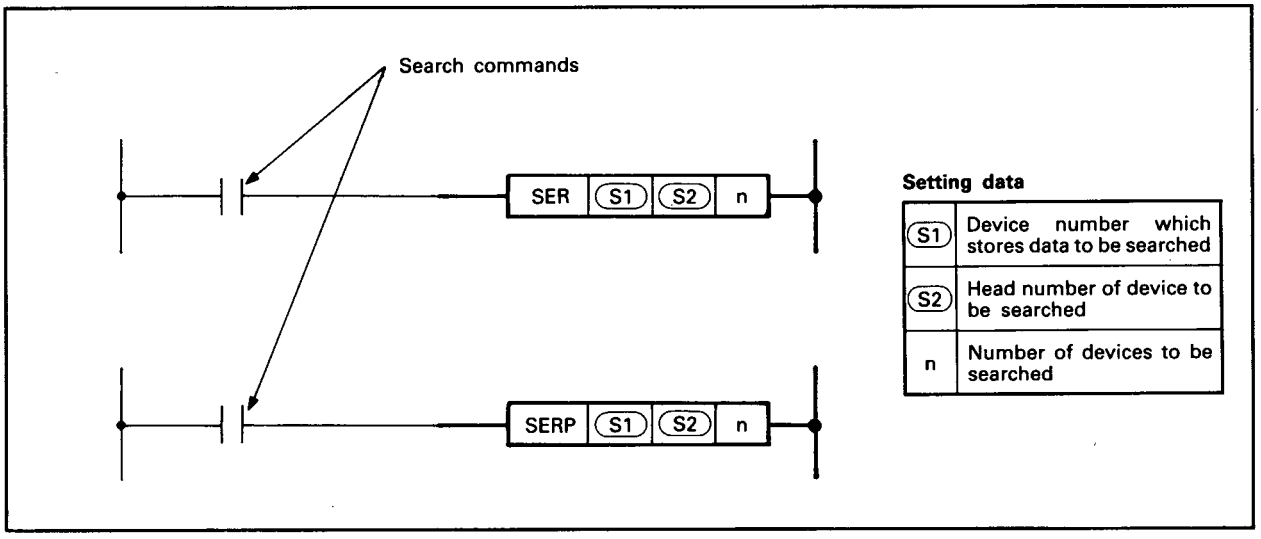
The data processing instructions perform operations such as the search, decode, and encode of data.

Classification	Instruction Symbol	Ref. Page
Search	SER	7-38 to 7-39
	SERP	7-38 to 7-39
Bit check	SUM	7-40 to 7-41
	SUMP	7-40 to 7-41
	DSUM	7-40 to 7-41
	DSUMP	7-40 to 7-41
Decode Encode	DECO	7-42 to 7-43
	DECOP	7-42 to 7-43
	ENCO	7-42 to 7-43
	ENCOP	7-42 to 7-43
7 segment decode	SEG	7-44 to 7-45
Bit set reset	BSET	7-46 to 7-47
	BSETP	7-46 to 7-47
	BRST	7-46 to 7-47
	BRSTP	7-46 to 7-47
16-bit data association/dissociation	DIS	7-48 to 7-50
	DISP	7-48 to 7-50
	UNI	7-48 to 7-50
	UNIP	7-48 to 7-50
ASCII conversion	ASC	7-51 to 7-52

7.4.1 16-bit data search (SER, SERP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(S1)								○	○	○	○	○	○	○	○	○	○	○										
(S2)								○	○	○	○	○											9		○		○	○
n																	○	○										



Setting data

(S1)	Device number which stores data to be searched
(S2)	Head number of device to be searched
n	Number of devices to be searched

Functions

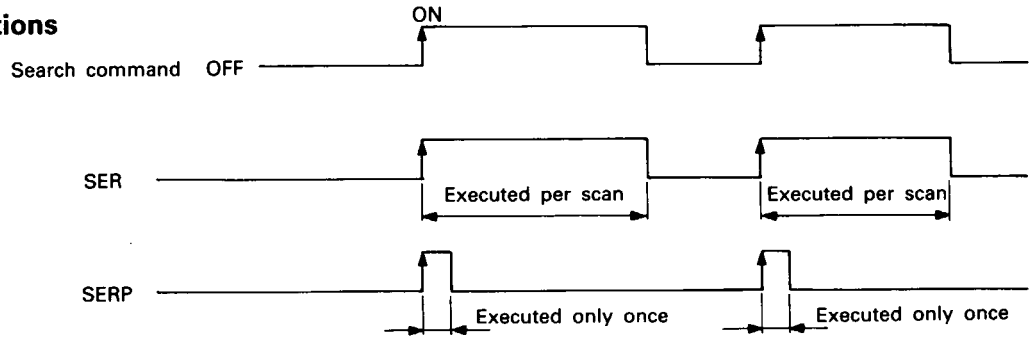
(1) Searches the data of "n" points, beginning with the 16-bit data of device specified at (S2), by use of the 16-bit data of device specified at (S1) as a keyword.

(2) Stores to A1 the number of data which have coincided with the keyword, and stores to A0 at which point from (S2) the first coinciding device number (relative value) is located.

(3) When "n" is negative, it is equal to 0.

(4) When "n" is 0, no processing is performed.

Execution Conditions



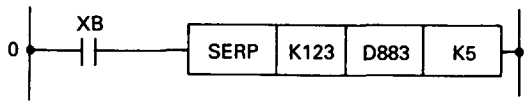
Operation Error

In the following case, operation error occurs and the error flag turns on.

○ When "n" points are searched beginning with (S2), the specified device range is exceeded.

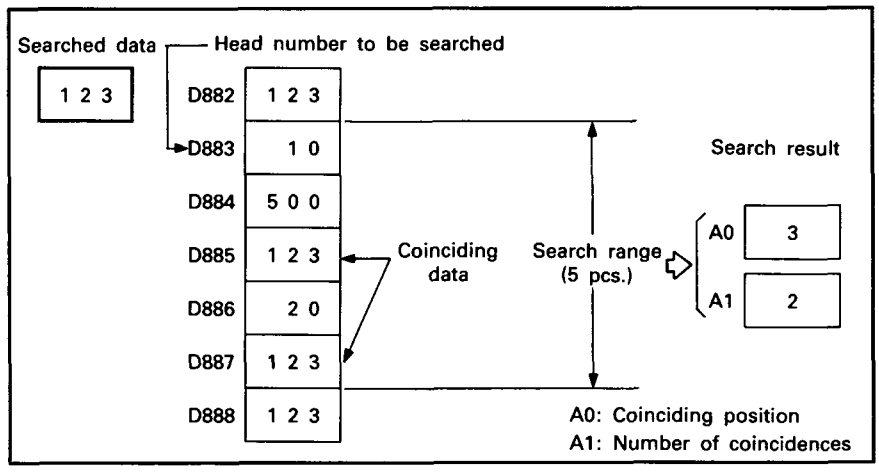
Program Example

Program which compares the data of D883 to 887 with 123 when XB turns on.



Coding

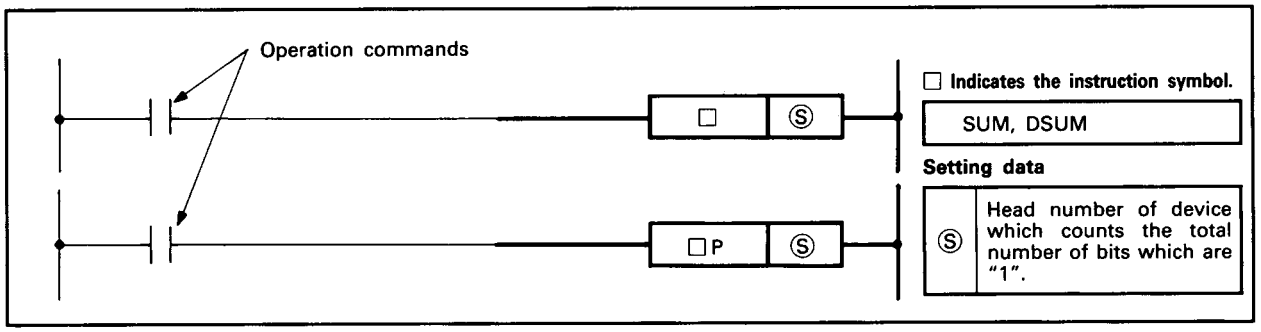
Step Number	Instruction	Device			
0	LD	XB			
1	SERP	K123	D883	K5	
10	END				



7.4.2 16-, 32-bit data bit check (SUM, SUMP, DSUM, DSUMP)

Processing Unit	Applicable CPU			
16/32 bits	A1N	A2N	A3N	A3H

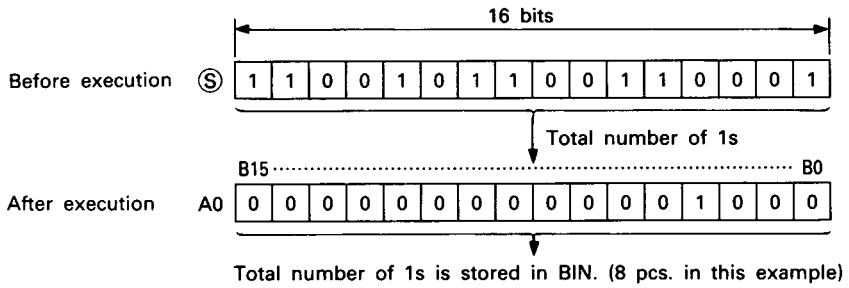
		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device								Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
SUM	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4	3		○		○	○
DSUM	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○		○							K1 to K8						



Functions

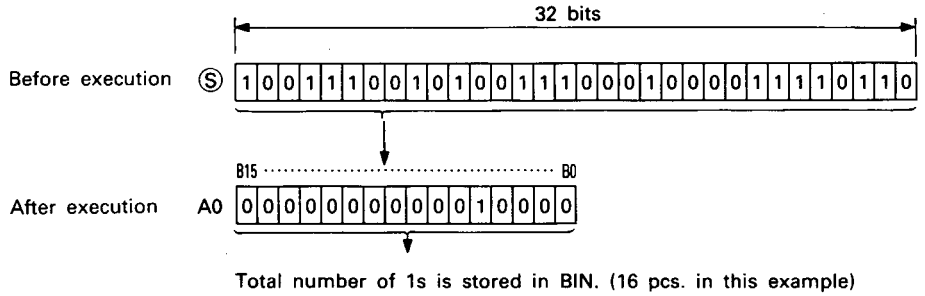
SUM

Stores in A0 the total number of bits which are one found in the 16-bit data of device specified at Ⓢ.

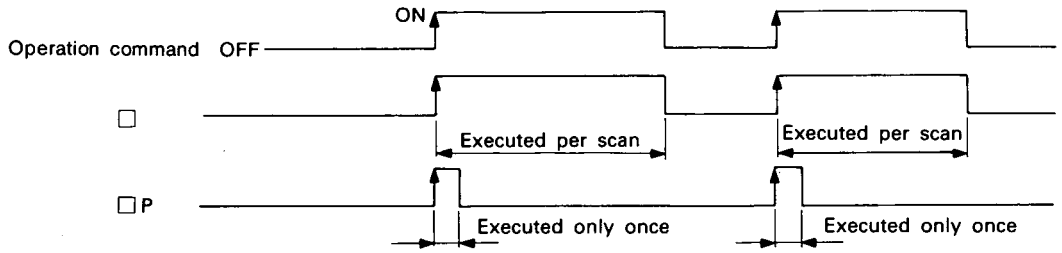


DSUM

Stores to A0 the total number of bits which are one found in the 32-bit data of device specified at Ⓢ.



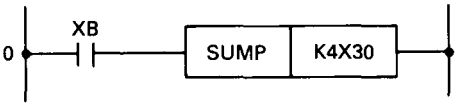
Execution Conditions



Program Examples

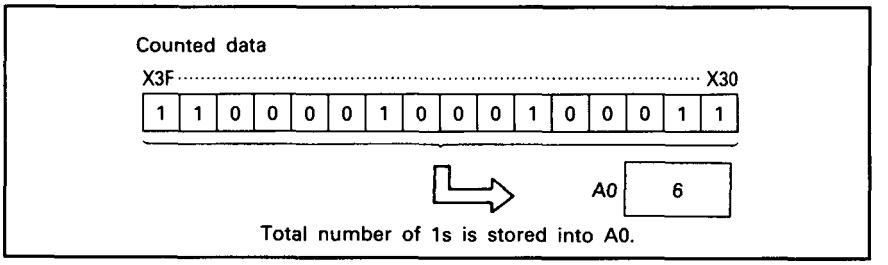
SUM

Program which obtains the number of bits, which are on (1), in the data of X30 to 3F when XB turns on.



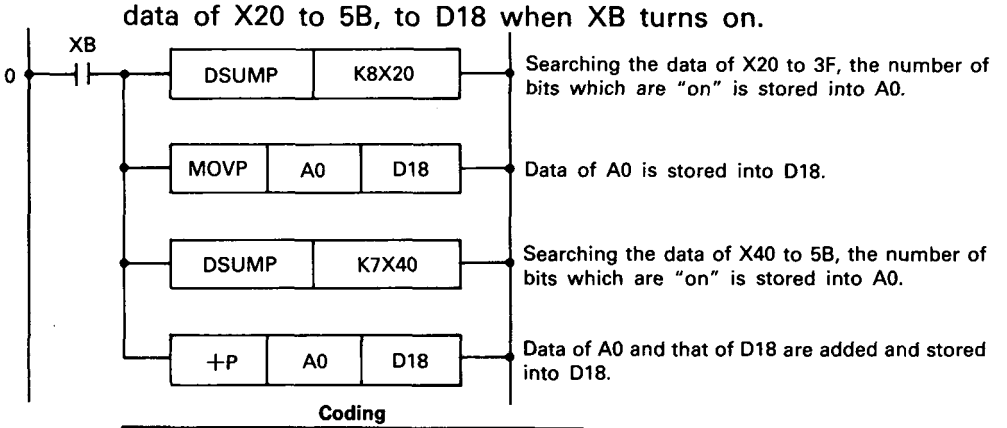
Coding

Step Number	Instruction	Device			
0	LD	XB			
1	SUMP	K4X30			
4	END				



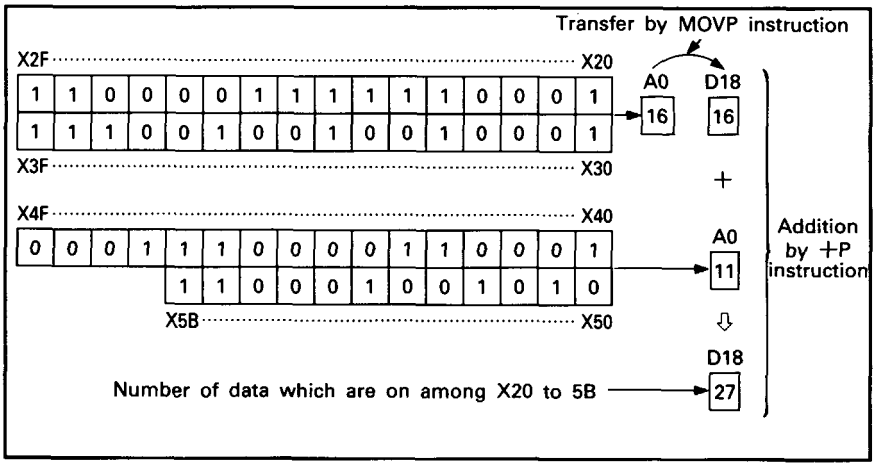
DSUM

Program which stores the number of bits, which are on (1), in the data of X20 to 5B, to D18 when XB turns on.



Coding

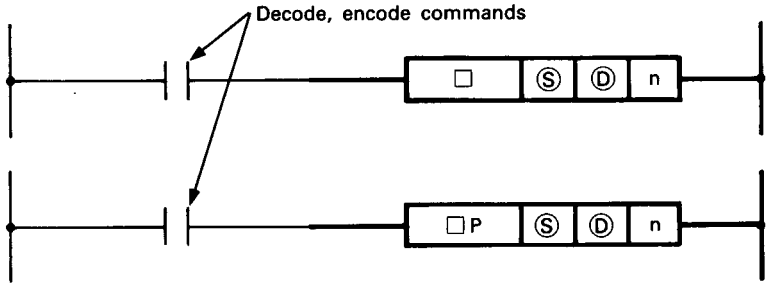
Step Number	Instruction	Device			
0	LD	XB			
1	DSUMP	K8X20			
4	MOVP	A0	D18		
9	DSUMP	K7X40			
12	+P	A0	D18		
17	END				



7.4.3 8 ↔ 256-bit decode, encode (DECO, DECOP, ENCO, ENCOP)

Processing Unit	Applicable CPU			
2^n bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
DECO	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○									
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○															
	n																	○	○									
ENCO	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○											
	Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○											
	n																	○	○									



□ Indicates the instruction symbol.

DECO, ENCO

Setting data

Ⓢ	Decode, encode data or head device number storing decode, encode data
Ⓓ	Head device number which will store the operation result
n	Effective bit length (1 to 8)

Functions

DECO

8 → 256 bit decode

(1) Decodes the lower "n" bits of device specified at Ⓢ and stores the result of decode data to 2^n bits which begin with the device specified at Ⓓ.

(2) For "n", 1 to 8 can be specified.

(3) When "n" is 0, no processing is performed and the contents of 2^n bits, which begin with the device specified at Ⓓ, do not change.

(4) A bit device is treated as one bit and a word device as 16 bits.

ENCO

256 → 8 bit decode

(1) Encodes the data of 2^n bits, which begin with Ⓢ, and stores the result to Ⓓ.

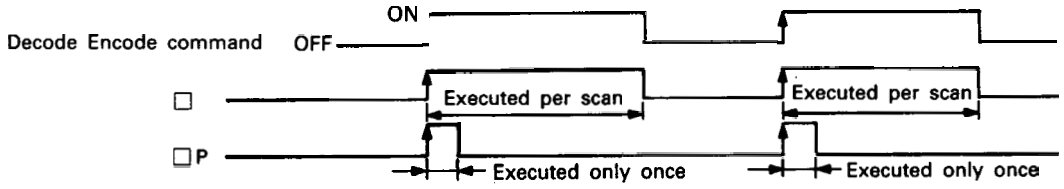
(2) For "n", 0 to 8 can be specified.

(3) When "n" is 0, no processing is performed and the contents of Ⓓ do not change.

(4) The bit device is treated as one bit and the word device as 16 bits.

(5) When multiple bits are 1, processing is performed for the last bit position.

Execution Conditions



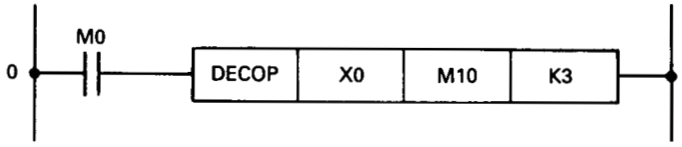
Operation Errors

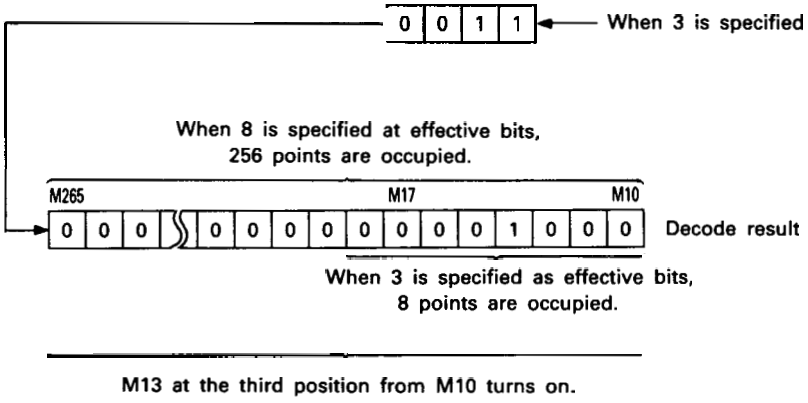
In the following case, operation error occurs and the error flag turns on.

- "n" in other than 0 to 8.
- 0 exists in all devices from S to 2n when the encode instruction is used.

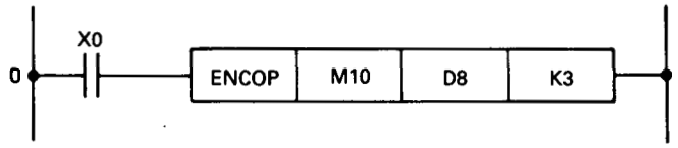
Program Examples

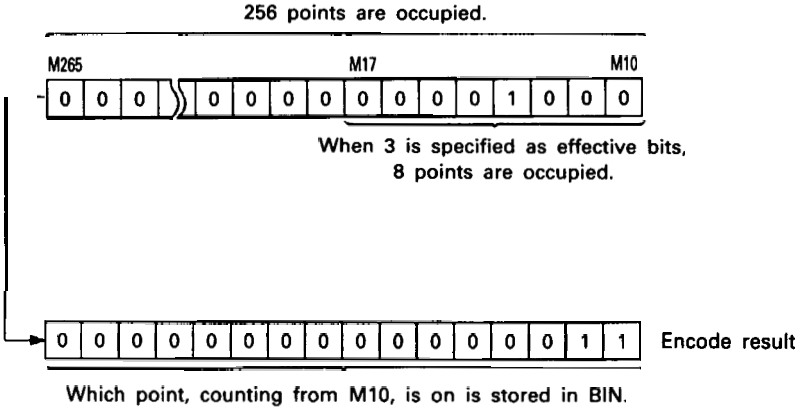
DECO





ENCO

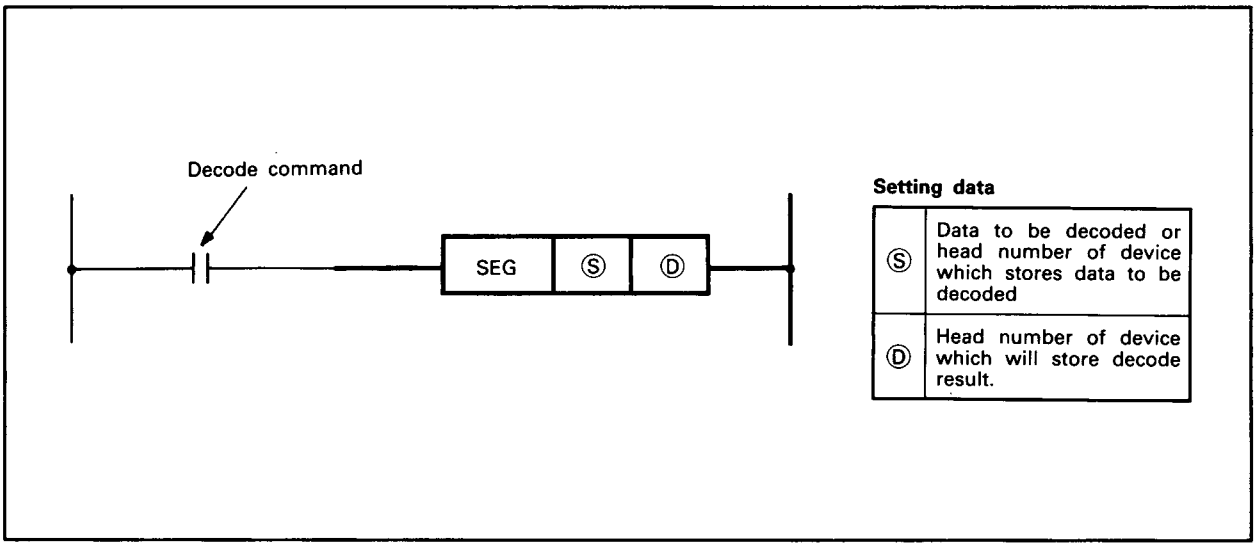




7.4.4 7 segment decode (SEG)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	7		○			
Ⓓ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○												



Setting data

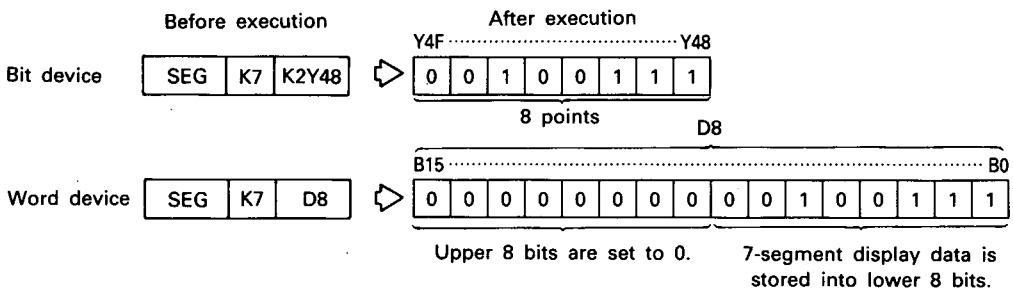
Ⓢ	Data to be decoded or head number of device which stores data to be decoded
Ⓓ	Head number of device which will store decode result.

Functions

(1) Decodes the data of 0 to F specified at the lower four bits of Ⓢ to seven-segment display data and stores the result to Ⓓ.

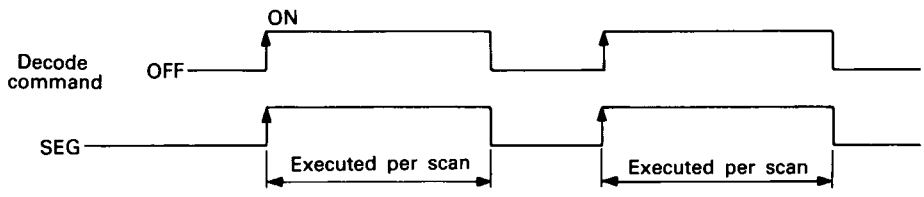
(2) When the device is a bit device (Y, M, L, S, B, F), indicates the head number of device which will store the seven-segment display data. When the device is a word device (T, C, D, R, A0, A1, Z, V), indicates the device number which will store the seven-segment display data.

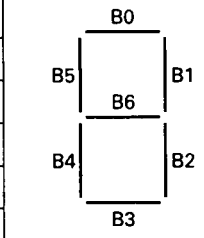
(3) The data is stored into the bit device and word device as shown below.



(4) For the seven-segment display data, refer to the next page.

Execution Conditions



Ⓢ		Configuration of 7-segment	Ⓓ								Displayed Data
Hexadecimal number	Bit pattern		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	0	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	b
C	1100		0	0	1	1	1	0	0	1	C
D	1101		0	1	0	1	1	1	1	0	d
E	1110		0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

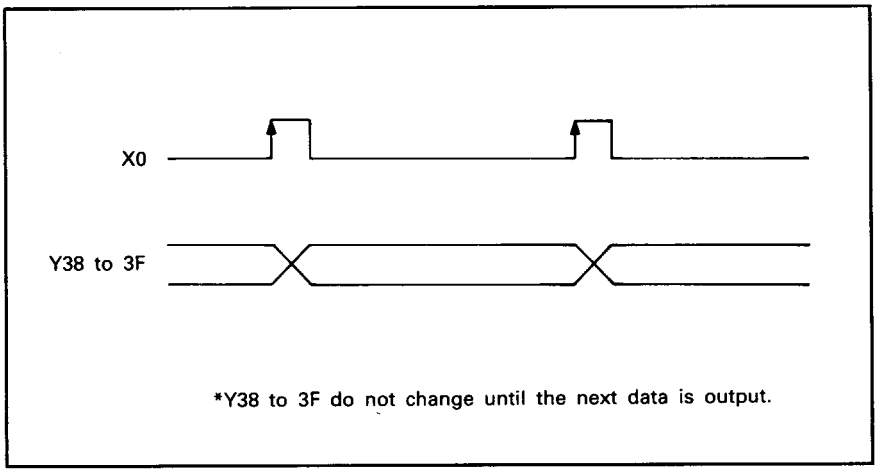
{ Head of bit device
{ The lowest bit of word device

Program Example

Program which converts the data of XC to F to seven-segment display data and sends the display data to Y38 to 3F when X0 turns on.

Coding

Step Number	Instruction	Device			
0	LD	X0			
1	SEG	K1XC	K2Y38		
8	END				

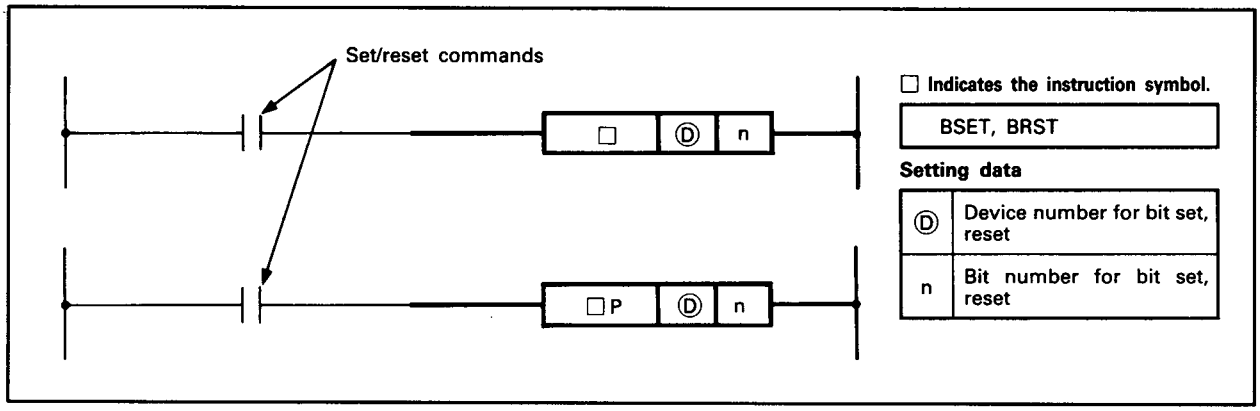


*Y38 to 3F do not change until the next data is output.

7.4.5 Word device bit set, reset (BSET, BSETP, BRST, BRSTP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ								○	○	○	○	○	○	○	○	○												
n																	○	○					7		○		○	○

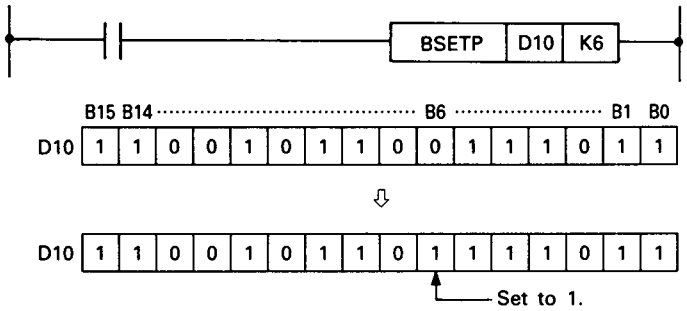


Functions

BSET

(1) Sets (1) the "n"th bit of word device specified at Ⓓ.

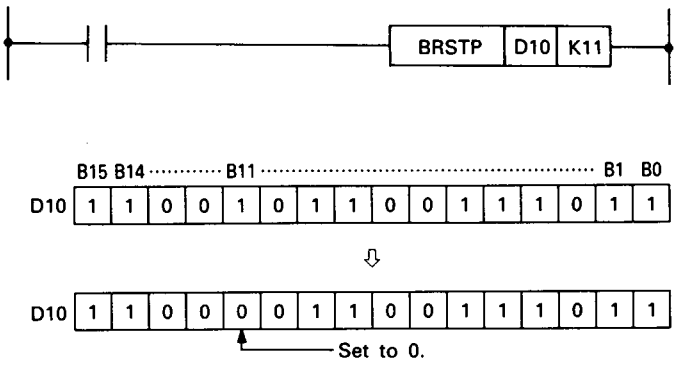
(2) For "n", 0 to 15 are effective. When 15 is exceeded, the instruction is executed at the lower four bits.



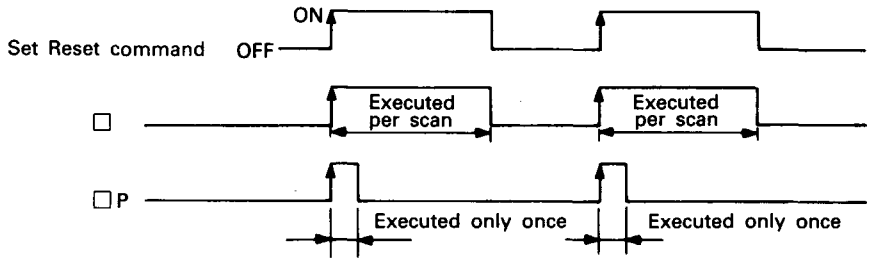
BRST

(1) Resets (0) the "n"th bit of word device specified at Ⓓ.

(2) For "n", 0 to 15 are effective. When 15 is exceeded, the instruction is executed at the lower four bits.



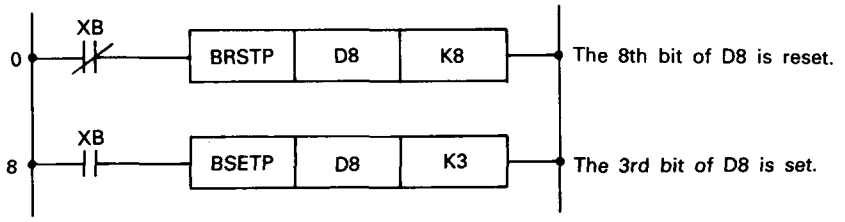
Execution Conditions



Program Example

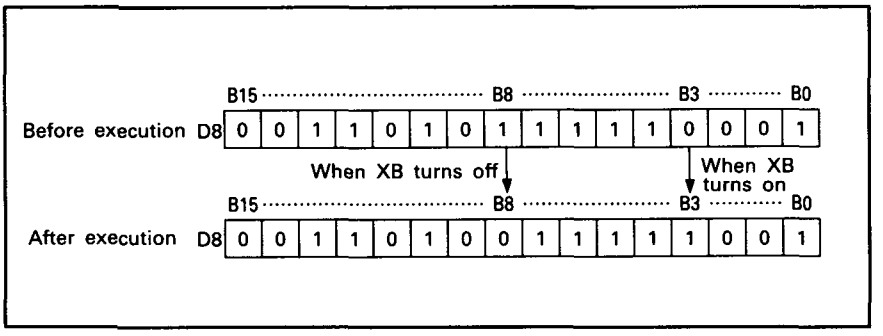
BEST , BRST

Program which sets the 3rd bit and 8th bit of D19 when X18 turns on.



Coding

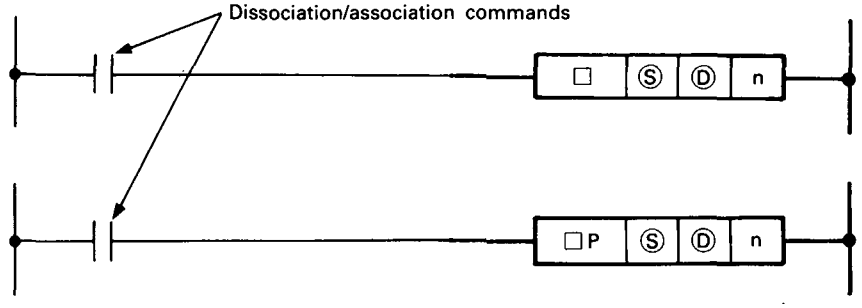
Step Number	Instruction	Device			
0	LDI	XB			
1	BRSTP	D8	K8		
8	LD	XB			
9	BSETP	D8	K3		
16	END				



7.4.6 16-bit data dissociation, association (DIS, DISP, UNI, UNIP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
DIS	Ⓢ								○	○	○	○	○	○	○	○	○	○	○				K1 to K4	9		○		○	○
	Ⓓ								○	○	○	○	○																
	n																	○	○										
UNI	Ⓢ								○	○	○	○	○																
	Ⓓ								○	○	○	○	○	○	○	○	○												
	n																	○	○										



□ Indicates the instruction symbol.

DIS, UNI

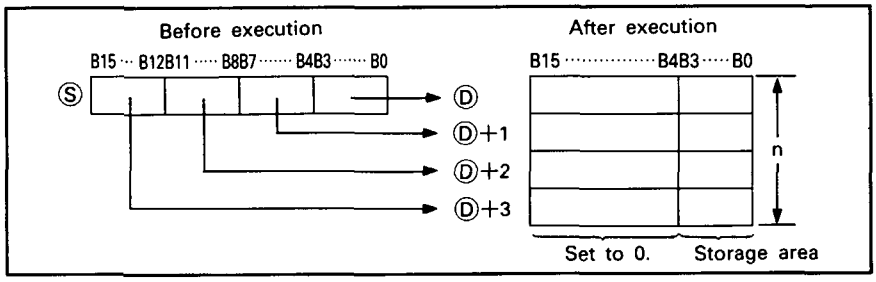
Setting data

Ⓢ	Head device number storing data to be dissociated, associated
Ⓓ	Device number which will store data dissociated, associated
n	• Number of devices for dissociation (1 to 4) • Number of data associated (1 to 4)

Functions

DIS

(1) Stores the data of lower "n" digits (one digit consists of four bits) of 16-bit data specified at Ⓢ into the lower four bits of devices of "n" points which begin with the device specified at Ⓓ.



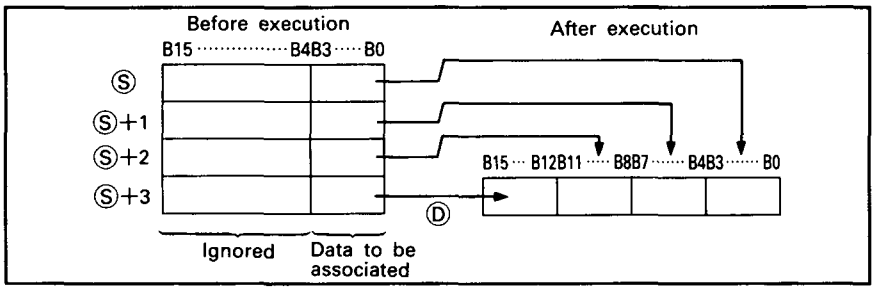
(2) The upper 12 bits of devices of "n" points, which begin with the device specified at Ⓓ, are set to 0.

(3) For "n", 1 to 4 can be specified.

(4) When "n" is 0, no processing is performed and the contents of "n" points beginning with the device of Ⓓ do not change.

UNI

(1) Associates the data of lower four bits of 16-bit data in devices of "n" points, which begin with the device specified at Ⓢ, to the 16-bit device specified at Ⓓ.

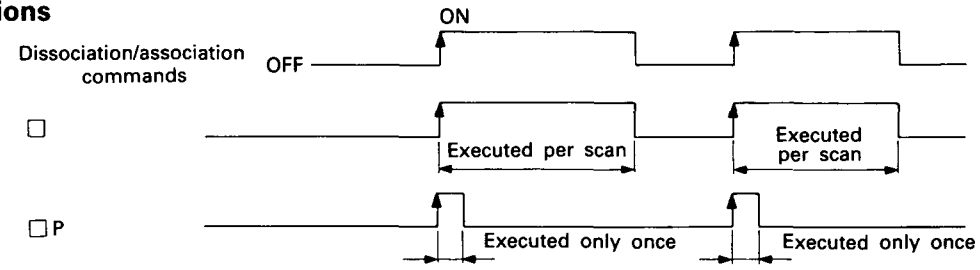


(2) The bits of upper (4 − n)-digits of device specified at Ⓓ, are set to 0.

(3) For "n", 1 to 4 can be specified.

(4) When "n" is 0, no processing is performed and the contents of device of Ⓓ do not change.

Execution Conditions



Operation Error

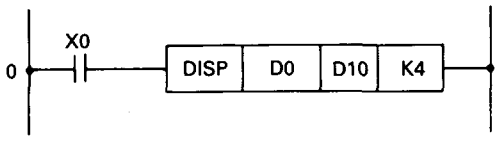
In the following case, operation error occurs and the error flag turns on.

○ "n" is other than 0 to 4.

Program Examples

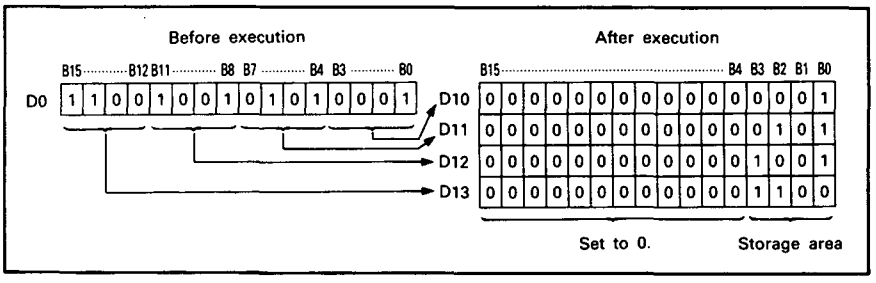
DIS

Program which stores the 16-bit data of D0 to the D10 to 13 per four bits when X0 turns on.



Coding

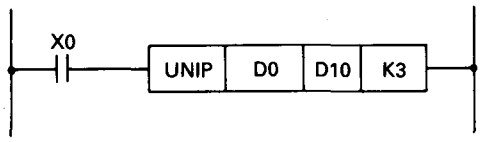
Step Number	Instruction	Device			
0	LD	X0			
1	DISP	D0	D10	K4	
10	END				



7

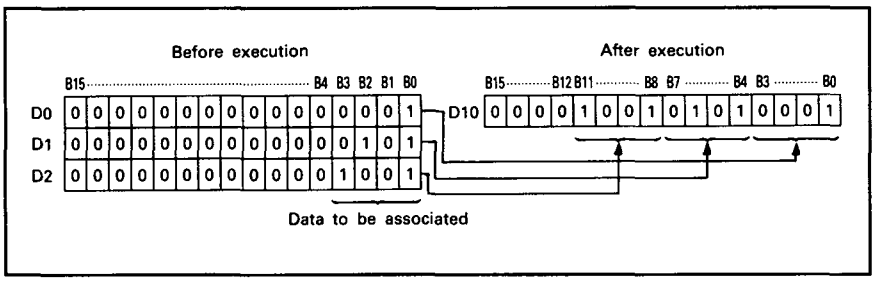
UNI

Program which stores the lower four-bit data of D0 to 2 to the D10 when X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	UNIP	D0	D10	K3	
10	END				



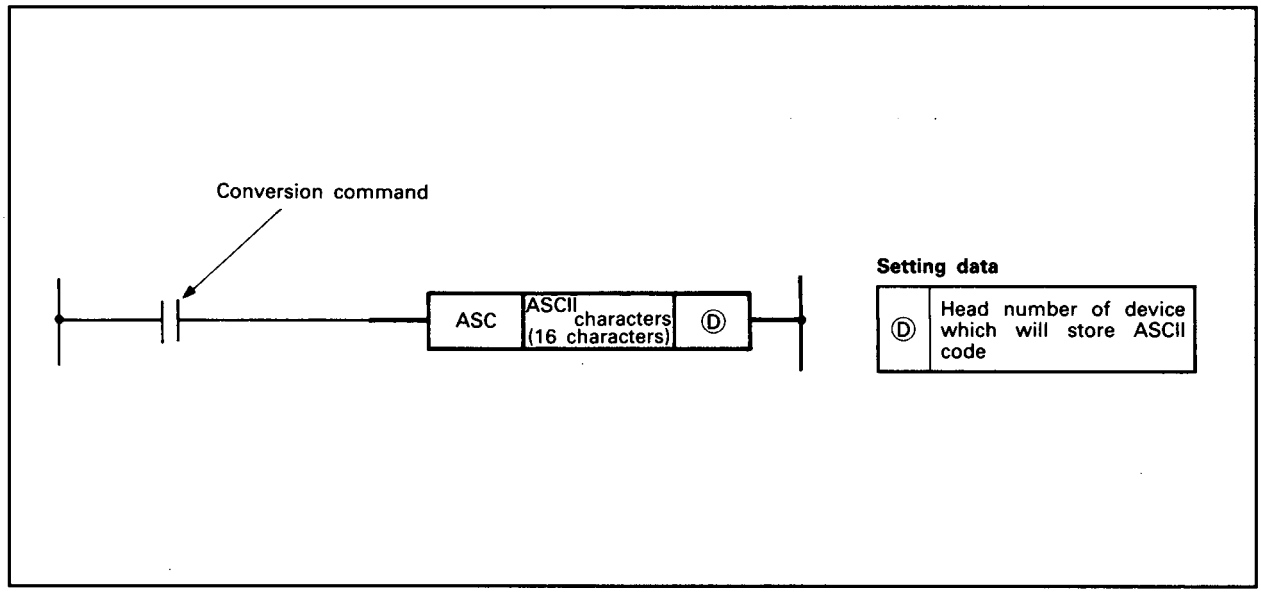
7

MEMO

7.4.7 ASCII code conversion (ASC)

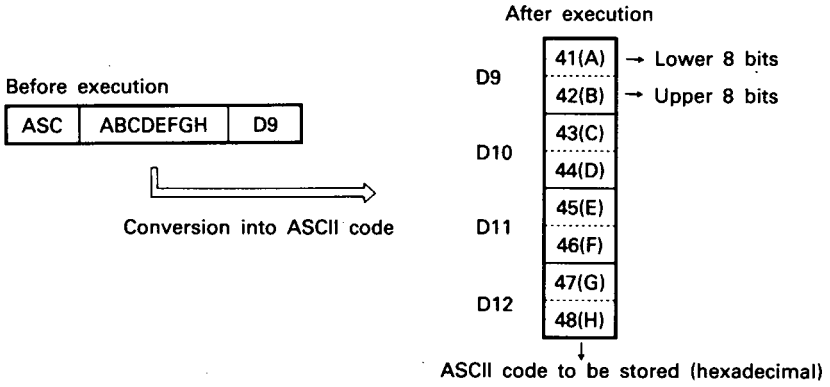
Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
Ⓓ								○	○	○	○	○											13		○		○	○

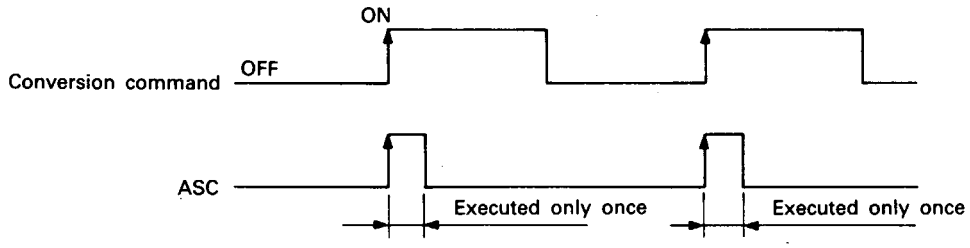


Function

Converts the specified alphanumeric characters into the ASCII code and stores the result into devices of four points which begin with the device specified at Ⓓ.

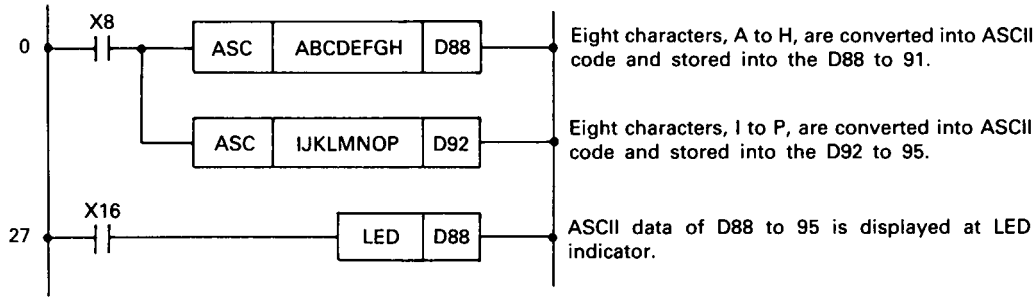


Executed Conditions



Program Example

Program which converts "ABCDEFGHIJKLMNOP" into the ASCII code and stores the result to the D88 to 95 when X8 turns on, and displays the ASCII data of D88 to 95 at the LED indicator on the front face of CPU when X16 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X8			
1	ASC	ABCDEFGH	D88		
14	ASC	IJKLMNOP	D92		
27	LD	X16			
28	LED	D88			
31	END				

MEMO

7.5 FIFO Instructions

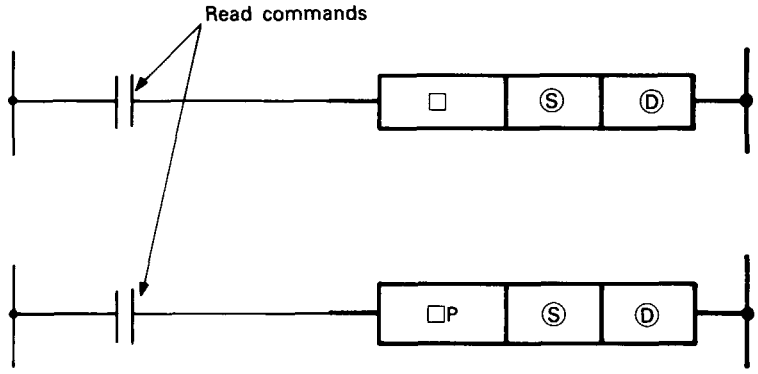
The FIFO instructions perform the write and read of data to and from the FIFO table.

Classification	Instruction Symbol	Ref. Page
Write	FIFW	7-54 to 7-57
	FIFWP	7-54 to 7-57
Read	FIFR	7-54 to 7-57
	FIFRP	7-54 to 7-57

7.5.1 FIFO table write, read (FIFW, FIFWP, FIFR, FIFRP)

Processing Unit	Applicable CPU			
16 bits	A1N	A2N	A3N	A3H

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
FIFW	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○				K1 to K4	7		○		○	○
	Ⓓ								○	○	○	○	○																
FIFR	Ⓢ		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○						K1 to K4						
	Ⓓ								○	○	○	○	○																



□ Indicates the instruction symbol.

FIFW, FIFR

Setting data

Ⓢ	• Data to be written to FIFO table or head number of device which stores data • Device number which will store data read
Ⓓ	Head device number of FIFO table

Functions

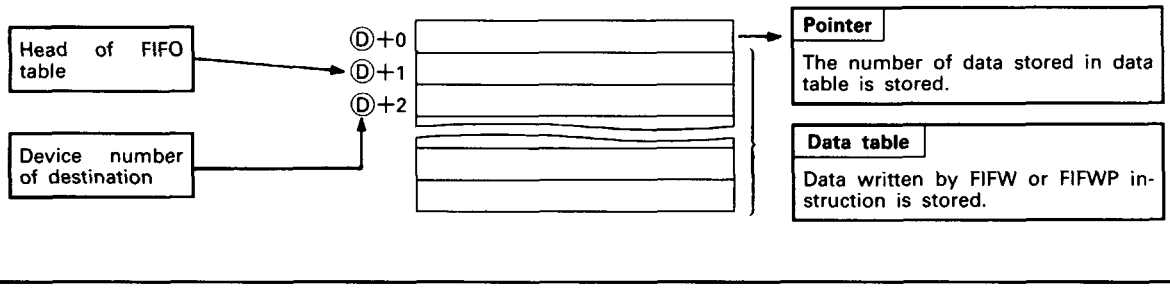
FIFW

(1) Performs the following actions:

1) Stores the data specified at Ⓢ into the data table of FIFO table. The storage position of data is as indicated below.

Data storage position =
head address of data table + content of pointer

2) Adds 1 to the content of pointer. (For the pointer, use the device specified at Ⓓ.)



(2) To perform the management of the number of data which may be written to multiple FIFO tables, use the user program.

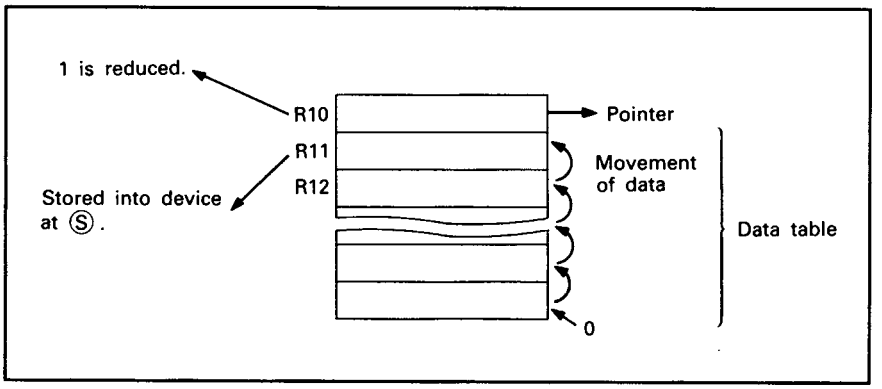
FIFR

(1) Reads data from the first device after the pointer of FIFO table and stores the data into the of Ⓢ.

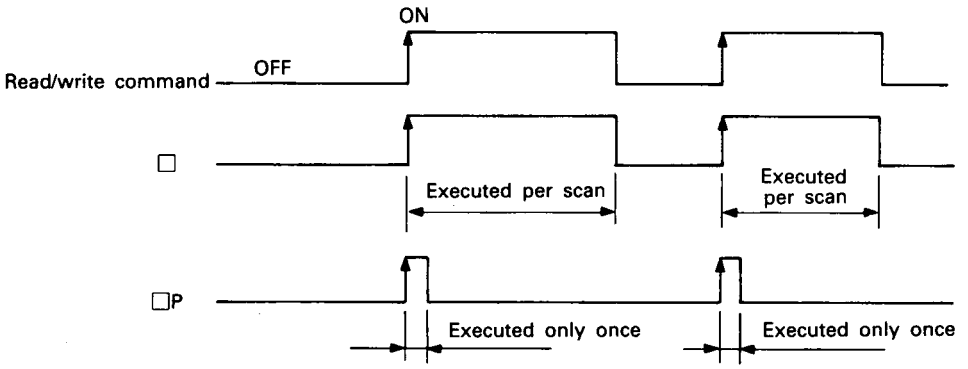
(2) The data of data table is shifted to the front one by one and the preceding data is set to 0. (i.e. data is lost)

(3) Subtracts 1 from the content of pointer.

(4) When the content of pointer is 0, no processing is performed.



Execution Conditions



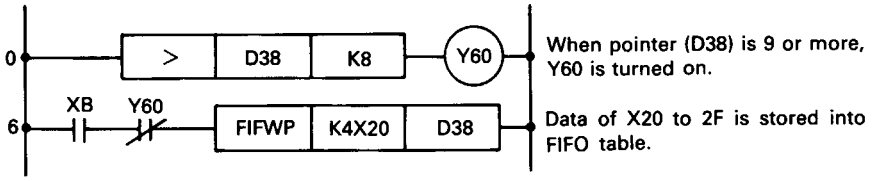
Operation Errors

In the following case, operation error occurs and the error flag turns on.

- (FIFO table head address) + (pointer) value exceeds the corresponding device range when the FIFW(P) instruction is used.
- The FIFR(P) instruction has been executed when the pointer value is 0.

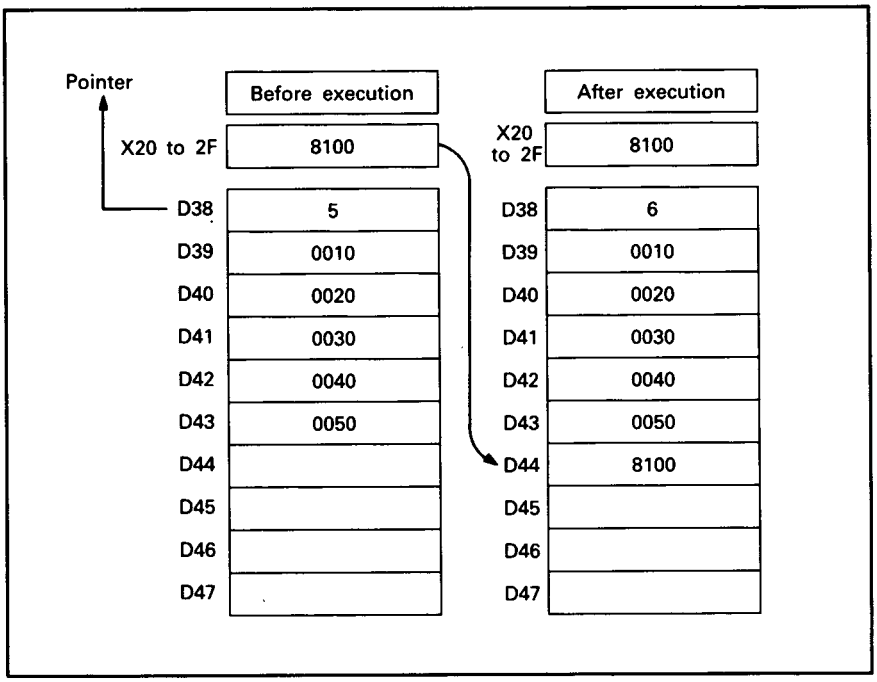
FIFW

Program which uses D38 to 47 as a FIFO table and temporarily stores the data of X20 to 2F when XB turns on. When the data exceeds 9, this program turns on Y60 to disable the execution of FIFW instruction.
(The data storage location is as shown below when the pointer value is 5.)



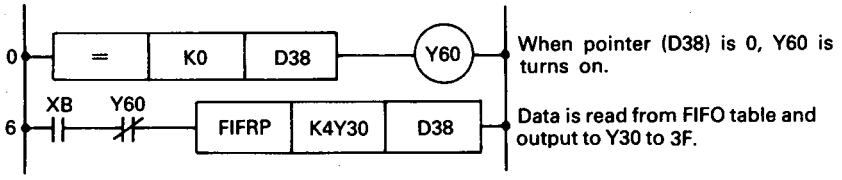
Coding

Step Number	Instruction	Device			
0	LD>	D38	K8		
5	OUT	Y60			
6	LD	XB			
7	ANI	Y60			
8	FIFWP	K4X20	D38		
15	END				



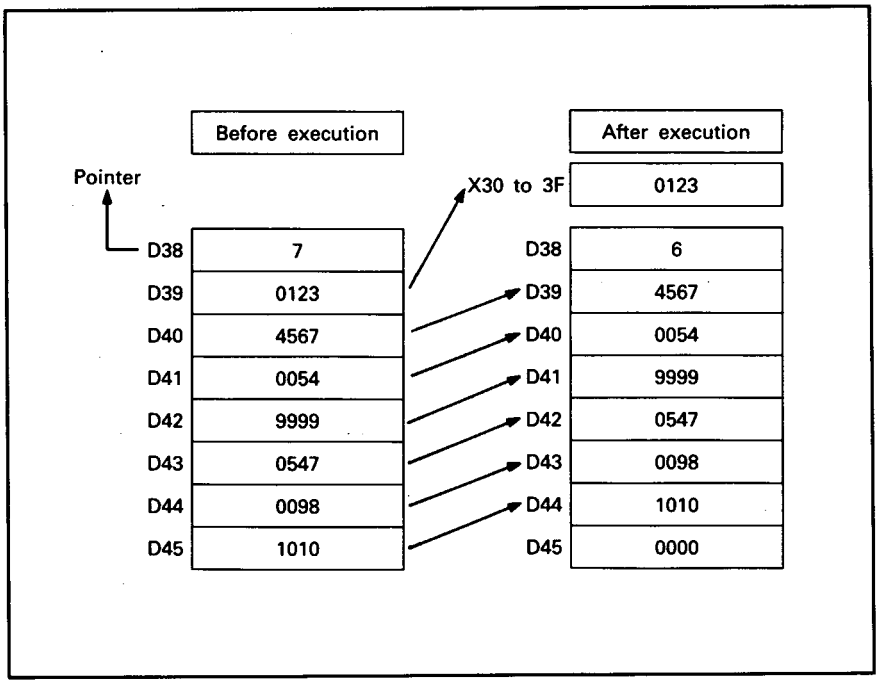
FIFR

Program which reads data from D38 to 45 of the FIFO table when XB turns on, and outputs the data to the Y30 to 3F.
(Data is read as shown below when the pointer value is 7.)



Coding

Step Number	Instruction	Device			
0	LD=	K0	D38		
5	OUT	Y60			
6	LD	XB			
7	ANI	Y60			
8	FIFRP	K4Y30	D38		
15	END				



7

MEMO

7.6 Buffer Memory Access Instructions

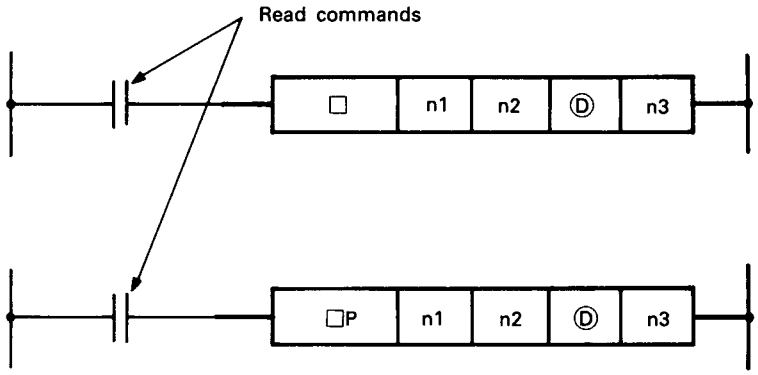
Buffer memory access instructions are used to transfer data between the CPU and special function module buffer memory.

Classification	Instruction Symbol	Ref. Page
Data read	FROM	7-59 to 7-60
	FROMP	7-59 to 7-60
	DFRO	7-59 to 7-60
	DFROP	7-59 to 7-60
Data write	TO	7-61 to 7-62
	TOP	7-61 to 7-62
	DTO	7-61 to 7-62
	DTOP	7-61 to 7-62

7.6.1 Special function module 1-, 2-word data read (FROM, FROMP, DFRO, DFROP)

Processing Unit	Applicable CPU			
1/2 word	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9010	M9011
n1																	○	○				K1 to K4 *	9		○	○	○
n2																	○	○									
Ⓓ	○*	○*	○*	○*	○*	○*	○*	○	○	○	○	○										* K1 to K8					
n3																	○	○									



□ Indicates the instruction symbol.

FROM, DFRO

Setting data

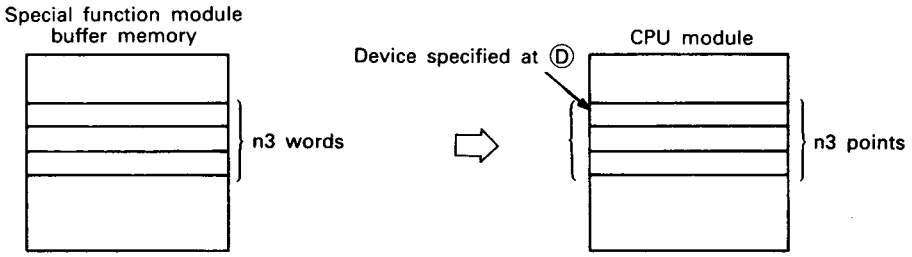
n1	Head I/O number of special function module
n2	Head address of data to be read
Ⓓ	Device number which will store read data
n3	Number of data to be read

*: Bit device may only be used with the A□NCPU.
• Digit specification: K1 to K4 for FROM.
K1 to K8 for DRO.

Functions

FORM

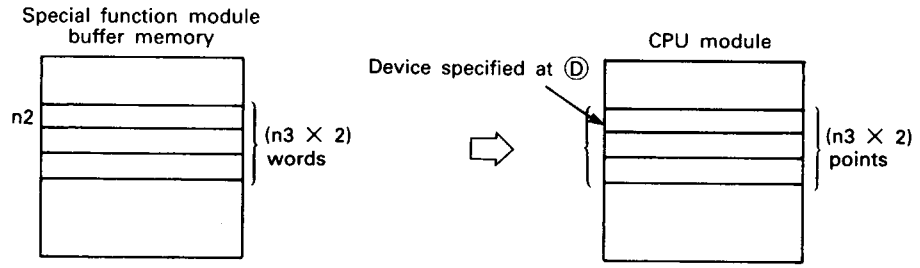
Reads the data of "n3" words, which start at the address specified at "n2" of buffer memory inside the special function module specified at "n1", and stores the data into devices which begin with the device specified at Ⓓ.



DERO

Reads the data of "n3 × 2" words, which start at the address specified at "n2" of buffer memory inside the special function module specified at "n1", and stores the data into devices which begin with the device specified at Ⓓ.

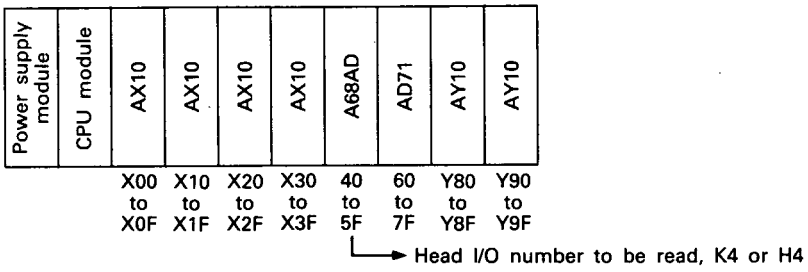
n2



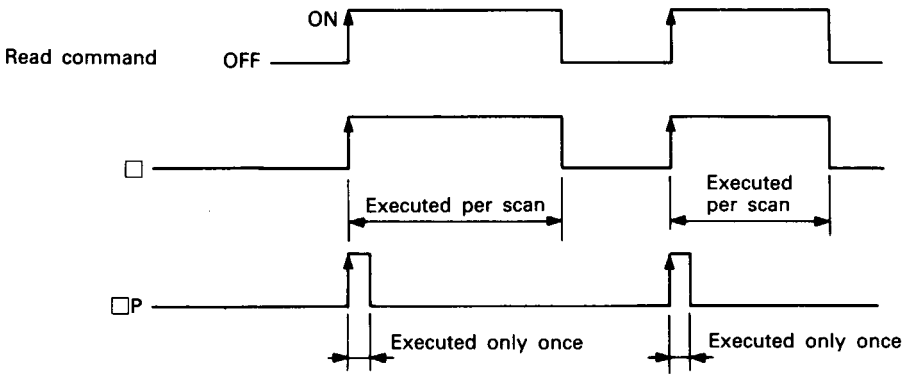
REMARKS

At "n1", specify the upper two digits of the head I/O number of slot where the special function module is loaded.

Example



Execution Conditions



Operation Errors

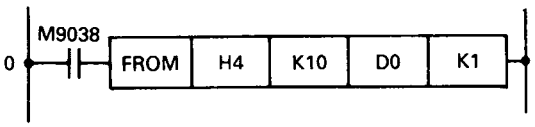
In the following cases, operation error occurs and the error flag turns on.

- Access cannot be made to the special function module.
- The I/O number specified at "n1" is not a special function module.
- "n3" points, which begin with the device specified at Ⓓ, exceeds the specified device range.

Program Examples

FROM

Program which reads the data of one word from the address 10 of buffer memory of A68AD, loaded in the slot 5 of the main base unit, to D0.

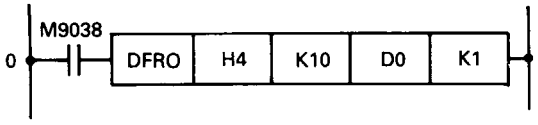


Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	FROM	H4	K10	D0	K1
10	END				

DFRO

Program which reads the data of two words from the address 10 of buffer memory of A68AD, loaded in the slot 5 of the main base unit, to D0 and 1.



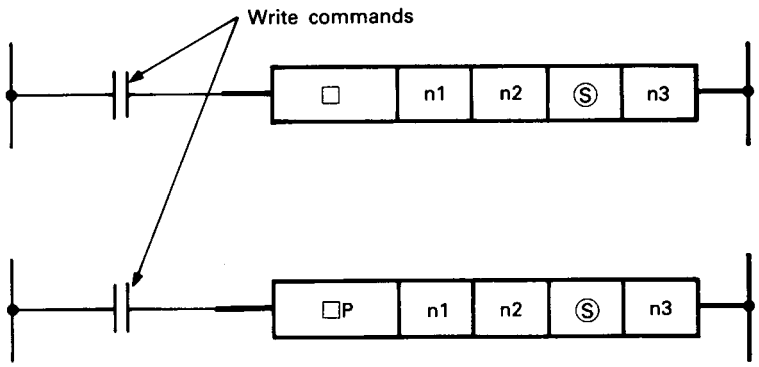
Coding

Step Number	Instruction	Device			
0	LD	M9038			
1	DFRO	H4	K10	D0	K1
10	END				

7.6.2 Special function module 1-, 2-word data write (TO, TOP, DTO, DTOP)

Processing Unit	Applicable CPU			
1/2 word	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n1																	○	○				K1 to K4 •	9					
n2																	○	○							○		○	○
Ⓢ	○*	○*	○*	○*	○*	○*	○*	○	○	○	○	○					○	○				• K1 to K8	11					
n3																	○	○										



□ Indicates the instruction symbol.

TO, DTO

Setting data

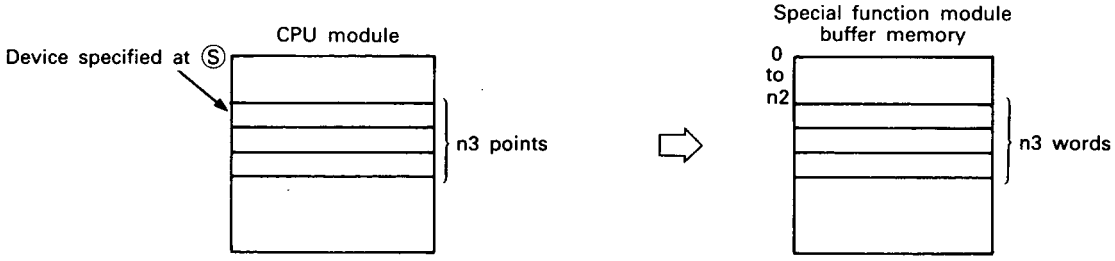
n1	Head I/O number of special function module
n2	Head address where data will be written
Ⓢ	Device number which stores data to be written
n3	Number of data to be written

*: Bit device may only be used with the A□NCPU.
• Digit specification: K1 to K4 for TO.
K1 to K8 for DTO.

Functions

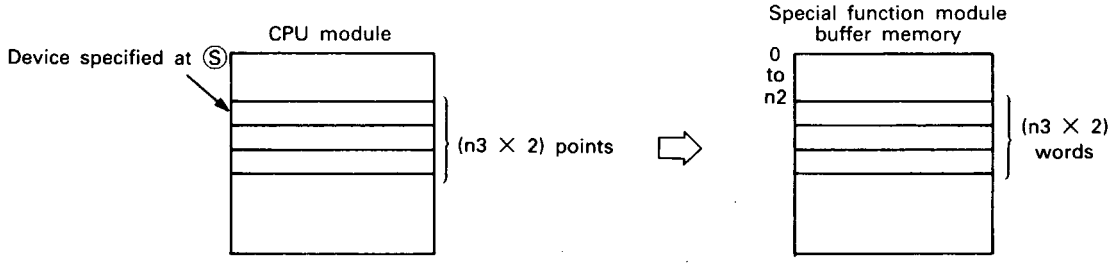
TO

Writes the data of "n3" points, which begin with the device specified at Ⓢ, to the addresses starting at the address specified at "n2" of buffer memory inside the special function module specified at "n1".



DTO

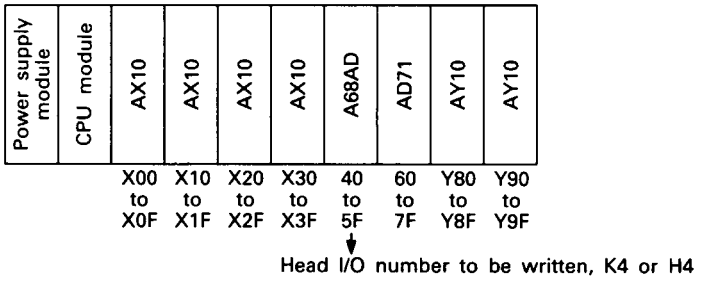
Writes the data of "n3 × 2" points, which begin with the device specified at Ⓢ, to addresses starting at the address specified at "n2" of buffer memory inside the special function module specified at "n1".



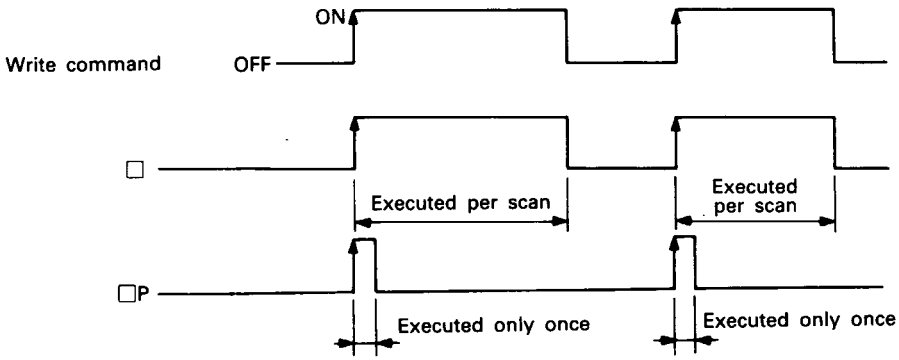
REMARKS

- At "n1", specify the upper two digits of the head I/O number of slot where the special function module is loaded.
- The number of steps is 11 when 2-word data is written by the DTO(P) instruction.

Example



Execution Conditions



Operation Errors

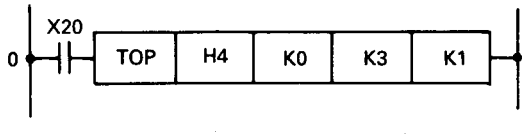
In the following cases, operation error occurs and the error flag turns on.

- ○ Access cannot be made to the special function module.
- ○ The I/O number specified at "n1" is not a special function module.
- ○ "n3" points, which begin with the device specified at Ⓓ, exceeds the specified device range.

Program Examples

TO

Program which sets three channels to the address 0 of buffer memory of A68AD, loaded in the slot 5 of the main base unit, when X20 turns on.

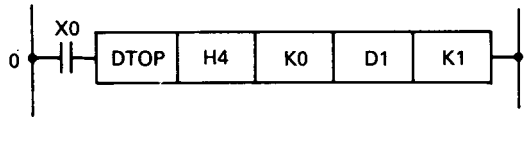


Coding

Step Number	Instruction	Device			
0	LD	X20			
1	TOP	H4	K0	K3	K1
10	END				

DTO

The following program writes D1 value to A68AD (loaded in slot 5 of the main base unit) buffer memory address 0 and D2 value to address 1 when X0 is switched on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	DTOP	H4	K0	D1	K1
12	END				

MEMO

7.7 Local, Remote I/O Station Access Instructions

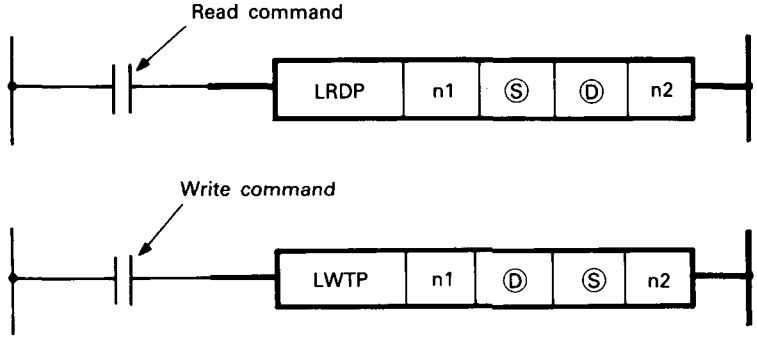
Local, remote I/O station access instructions are used to transfer data in a data link system.

Classification		Instruction Symbol	Ref. Page
Local station	Read	LRDP	7-64 to 7-67
	Write	LWTP	7-64 to 7-67
Remote I/O station	Read	RFRP	7-68 to 7-70
	Write	RTOP	7-68 to 7-70

7.7.1 Local station data read, write (LRDP, LWTP)

Processing Unit	Applicable CPU			
1 word	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n1																	○	○									
Ⓢ								○	○	○	○												11		○		○	○
Ⓓ								○	○	○	○																	
n2																	○	○									



Setting data

n1		Local station number
Ⓢ	LRDP	Head device number of local station to be read
	LWTP	Head device number of master station which stores data to be written
Ⓓ	LRDP	Head device number of master station which will store read data
	LWTP	Head device number of local station where data will be written
n2		Number of data read/written (1 to 32)

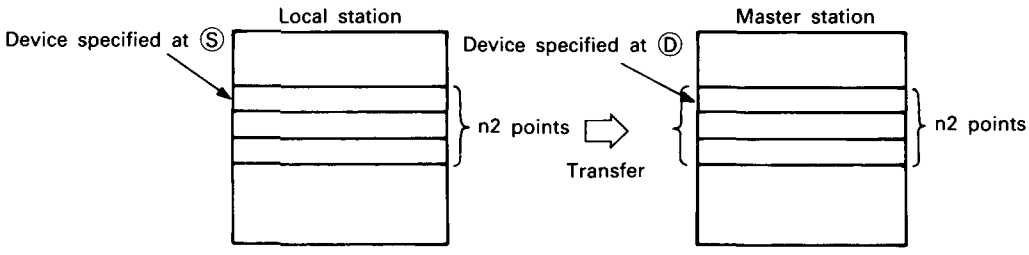
Functions

LRDP

(1) Stores the data of "n2" points, which begin with the device specified at Ⓢ of local station specified at "n1", to devices, which begin with the device specified at Ⓓ, of master station.

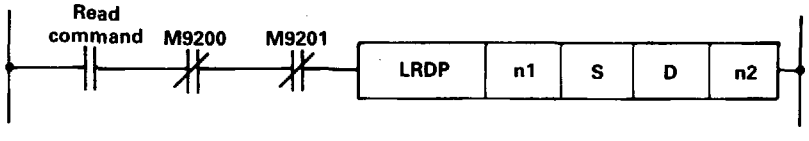
(2) When the LRDP instruction is executed, M9200 turns on and data is read from the local station. When the data is stored into "n2" points which begin with the device specified at Ⓓ, M9201 turns on.

(3) After the completion of LRDP instruction, M9200 and 9201 remain on. Therefore, turn them off by the sequence program.



POINT

Provide interlock by use of M9200 and 9201 so that the LRDP instruction is not executed again during the read operation of data from the local station by the LRDP instruction.

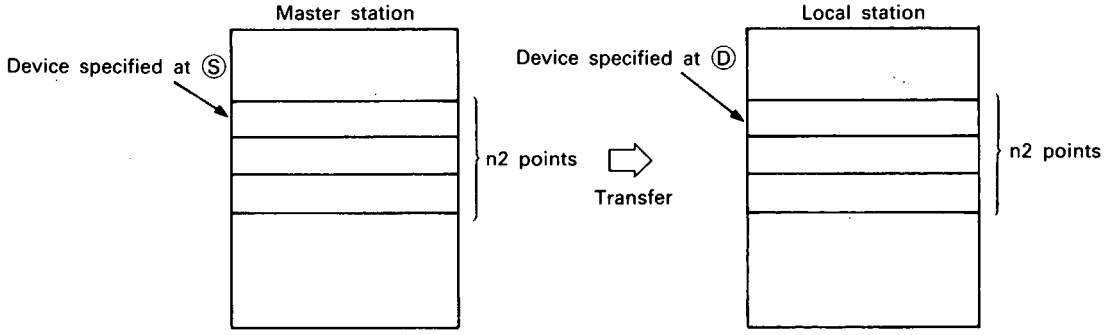


LWTP

(1) Stores the data of "n2" points, which begin with the device specified at Ⓢ of master station, to devices, which begin with the device specified at Ⓓ, of local station specified at "n1".

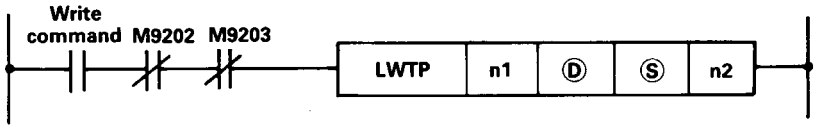
(2) When the LWTP instruction is executed, M9202 turns on. When data is written from the master station to the local station, M9203 turns on.

(3) After the completion of LWTP instruction, M9202 and 9203 remain on. Therefore, turn them off by the sequence program.

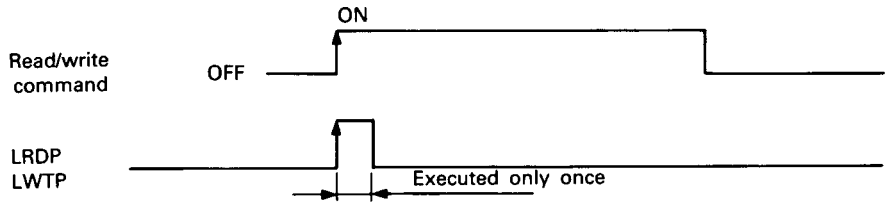


POINT

Provide interlock by use of M9202 and 9203 so that the LWTP instruction is not executed again during the write operation of data to the local station by the LWTP instruction.



Execution Conditions



Operation Errors

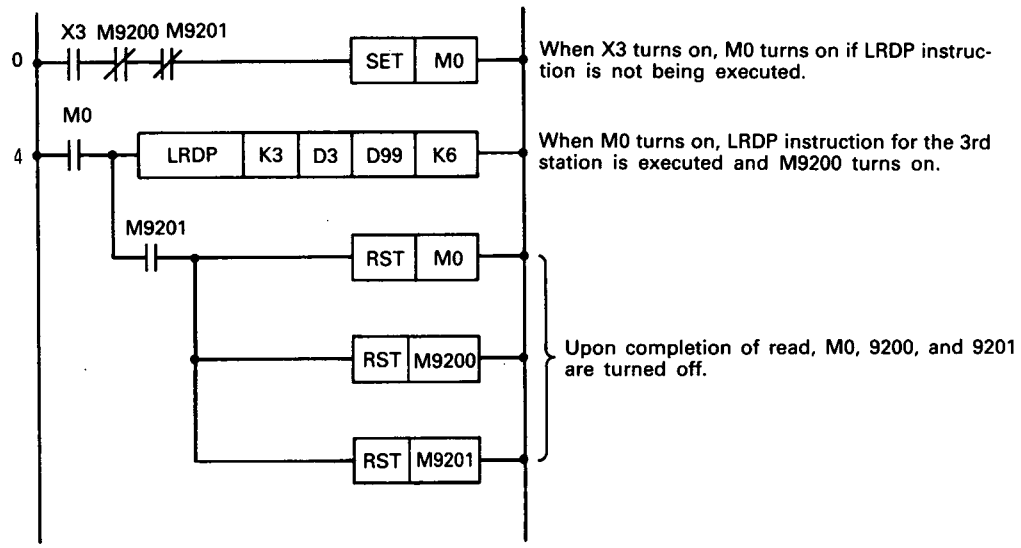
In the following cases, operation error occurs and the error flag turns on.

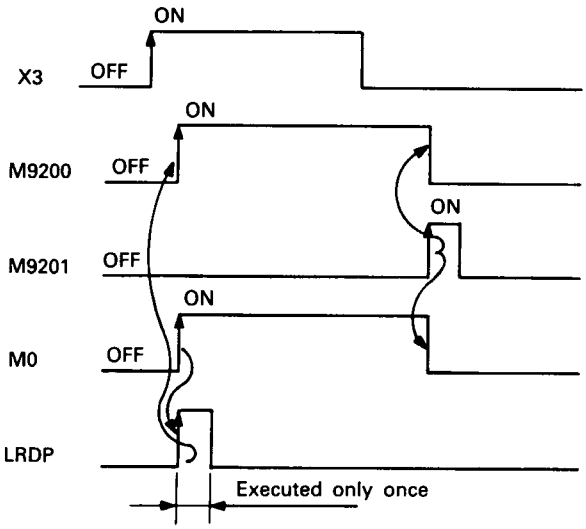
- The station number specified at "n1" is not a local station.
- "n2" points starting at Ⓢ or "n2" points starting at Ⓓ exceeds the specified device range.
- Specification of "n2" is other than 1 to 32.

Program Examples

LRDP

Program which stores the data of D3 to D8 of the 3rd local station into the D99 to 104 of master station when M3 turns on.





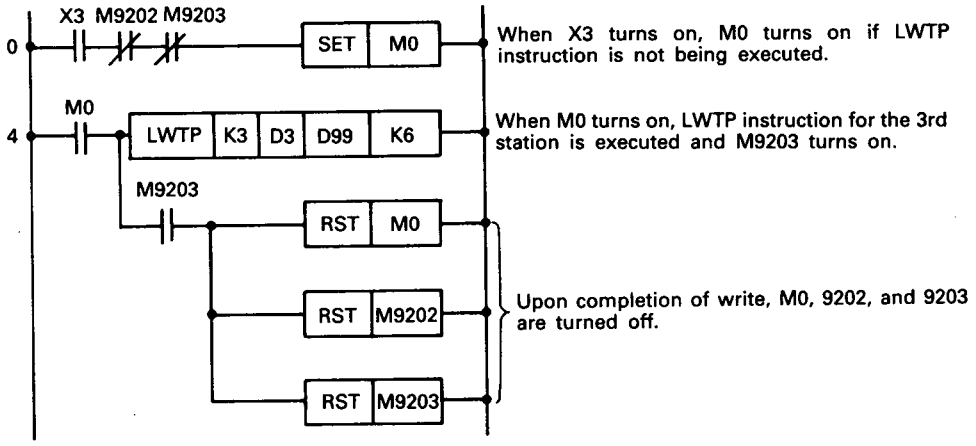
Coding

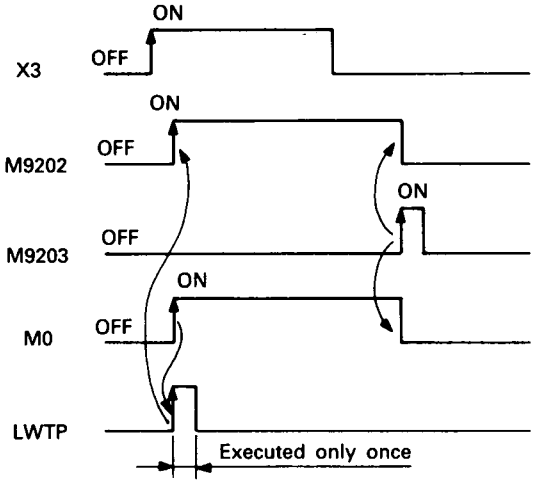
Step Number	Instruction	Device			
0	LD	X3			
1	ANI	M9200			
2	ANI	M9201			
3	SET	M0			
4	LD	M0			
5	LRDP	K3	D3	D99	K6
16	AND	M9201			
17	RST	M0			
18	RST	M9200			
21	RST	M9201			
24	END				

7

LWTP

Program which stores the data of D99 to 104 of master station to the D3 to D8 of the 3rd local station when X3 turns on.





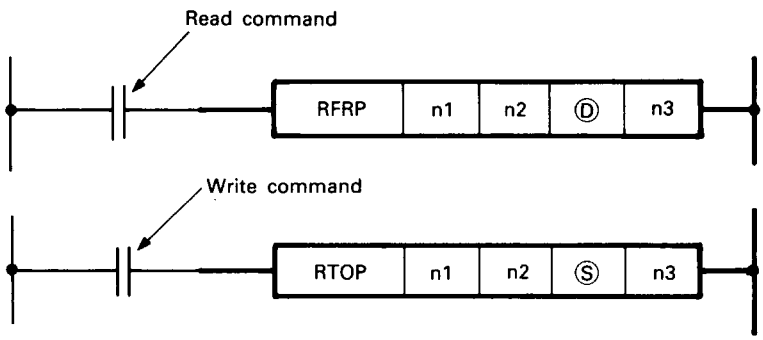
Coding

Step Number	Instruction	Device			
0	LD	X3			
1	ANI	M9202			
2	ANI	M9203			
3	SET	M0			
4	LD	M0			
5	LWTP	K3	D3	D99	K6
16	AND	M9203			
17	RST	M0			
18	RST	M9202			
21	RST	M9203			
24	END				

7.7.2 Remote I/O station data read, write (RFRP, RTOP)

Processing Unit	Applicable CPU			
1 word	A1N	A2N	A3N	A3H

	Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device								Constant		Pointer		Level						
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n1																	○	○										
n2																	○	○										
Ⓢ											○												11		○		○	○
Ⓓ											○																	
n3																	○	○										



Setting data

n1		Head I/O number of special function module specified by master station
n2	RFRP	Head address inside intelligent module which stores data to be read
	RTOP	Address inside special function module where data will be written
Ⓓ	RFRP	Head number of link register which will store read data
Ⓢ	RTOP	Head number of link register which stores data to be written
n3		Number of data read/written (1 to 16)

Functions

RFRP

(1) Stores the data of "n3" points from the address specified at "n2" inside the special function module, which has the I/O number specified at "n1", to link registers which begin with the one specified at Ⓓ of master station.

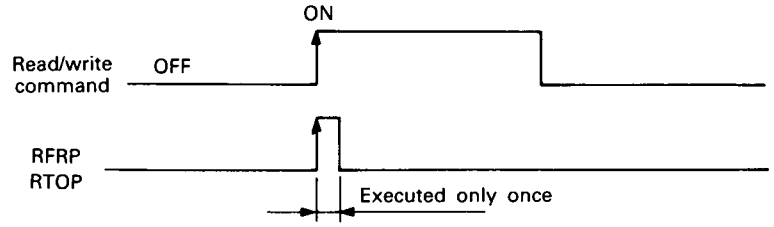
(2) When the RFRP instruction is executed, Yn1E of the specified special function module turns on. After the completion of read operation, X(n1 + 1)E turns on during two scans of link.

RTOP

(1) Stores the data of "n3" points from the link registers specified at Ⓢ to the addresses, which start at the address specified at "n2", inside the special function module of remote station which has the I/O number specified at "n1".

(2) When the RTOP instruction is executed, Yn2F of the specified special function module turns on. After the completion of write operation, X(n2 + 1)F turns on during two scans of link.

Execution Conditions



Operation Errors

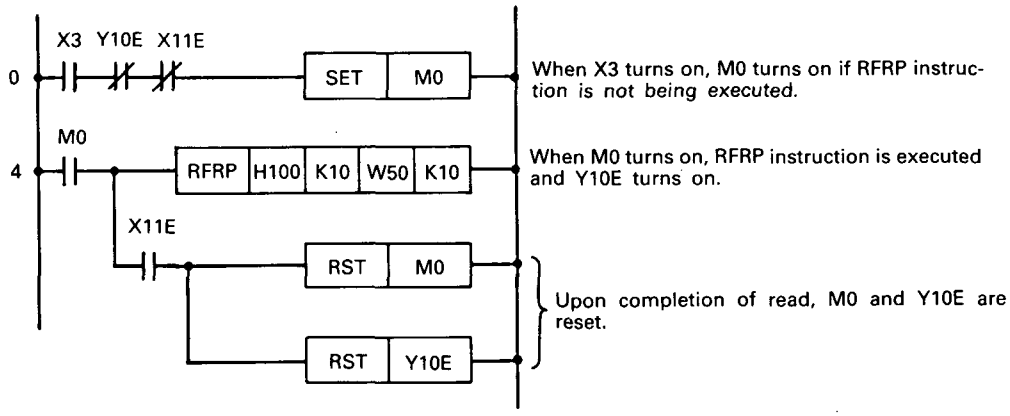
In the following cases, operation error occurs and the error flag turns on.

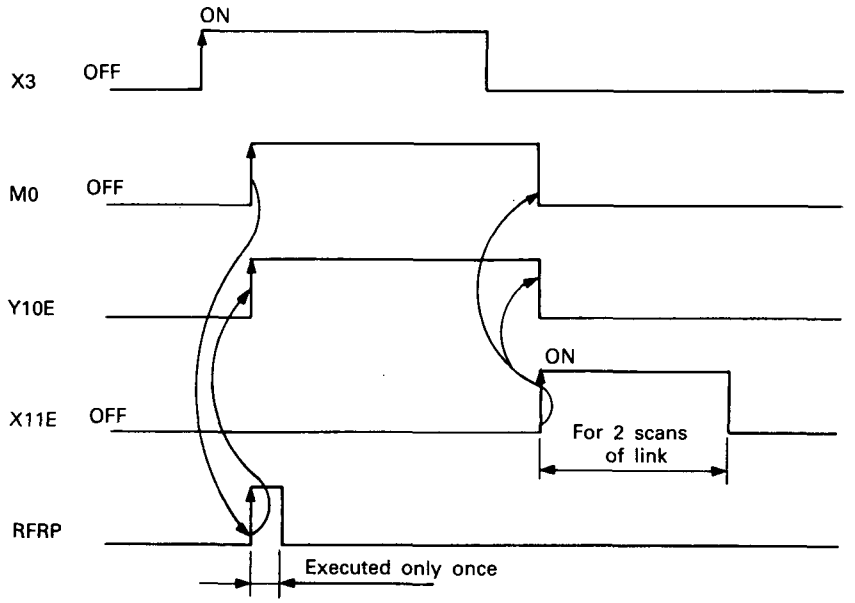
- ○ The specified station is not a remote station.
- ○ The head I/O number specified at "n1" is not a special function module.
- ○ The number of points, n3, exceeds the link register range (W0 to 3FF).

Program Examples

RFRP

Program which reads the data of 10 points from the address 10 of A68AD loaded in the slot of remote station, of which I/O numbers are 100 to 11F, to the W50 to 59 when M3 turns on.





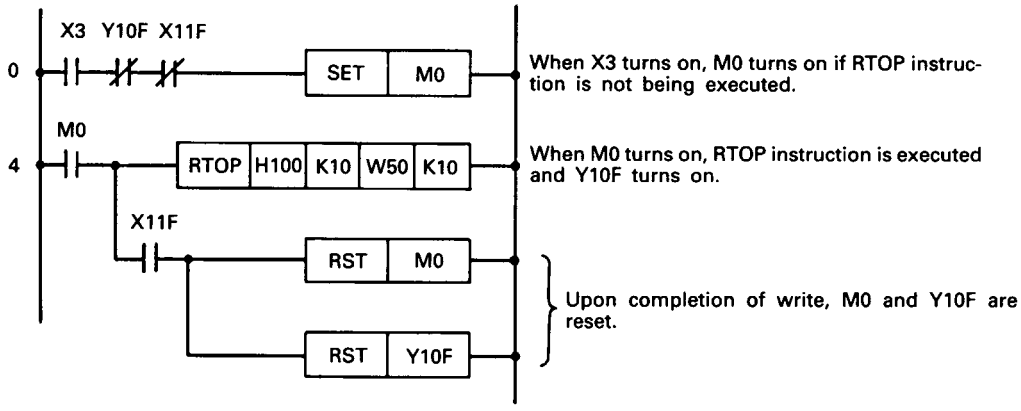
Coding

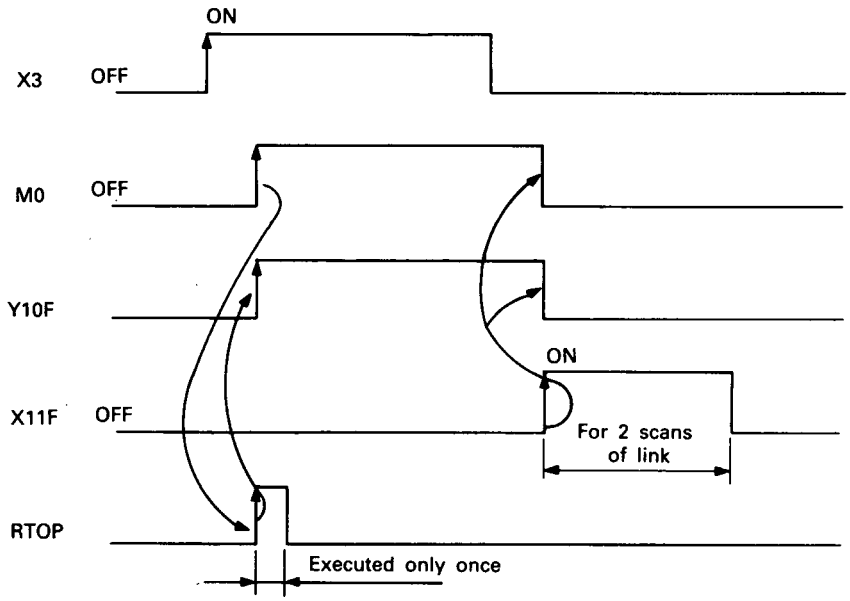
Step Number	Instruction	Device			
0	LD	X3			
1	ANI	Y10E			
2	ANI	X11E			
3	SET	M0			
4	LD	M0			
5	RFRP	H100	K10	W50	K10
16	AND	X11E			
17	RST	M0			
18	RST	Y10E			
19	END				

Program Examples

RTOP

Program which writes the data of W50 to 59 to the 10 points starting at the address 10 of A68AD loaded in the slot of remote station, of which I/O numbers are 100 to 11F, when X3 turns on.





Coding

Step Number	Instruction	Device			
0	LD	X3			
1	ANI	Y10F			
2	ANI	X11F			
3	SET	M0			
4	LD	M0			
5	RTOP	H100	K10	W50	K10
16	AND	X11F			
17	RST	M0			
18	RST	Y10F			
19	END				

MEMO

7.8 Display Instructions

(1) Display instructions are used to output ASCII codes to the output module, indicate data on the LED display, and reset the annunciator.

(2) The display instructions are available in the following five types.

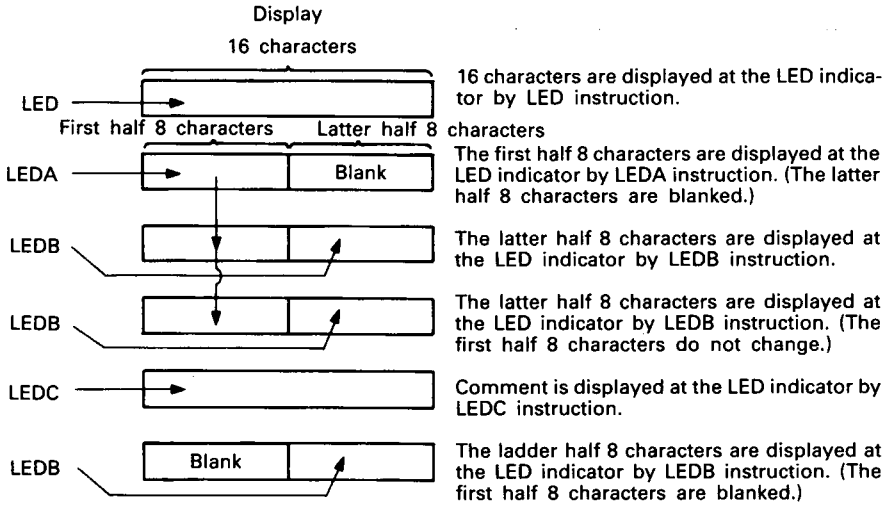
Classification	Instruction Symbol	Ref. Page
ASCII code output	PR	7-73 to 7-78
	PRC	7-73 to 7-78
Display	LED	7-79 to 7-81
	LEDC	7-79 to 7-81
	LEDA	7-82 to 7-83
	LEDB	7-82 to 7-83
Display reset	LEDR	7-84 to 7-85

(3) The priority of display at the LED indicator is as indicated below.

Priority: High 1) Display due to self-diagnostic error
2) Display of annunciator (F) number
3) Display due to LED, LEDC, LEDA, or LEDB
Low 4) BATTERY ERROR

(4) When there is a display at the LED indicator due to 1 to 3, the execution of display instruction does not change the display. When there is a display at the LED indicator due to 5, the execution of display instruction provides the display of that display instruction.

(5) When the display instruction is executed, the display is as shown below.



IB (NA) 66147-A

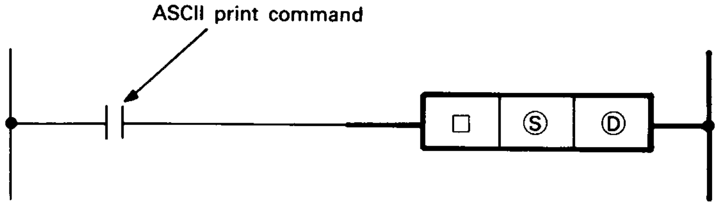
(6) Those which can be displayed at the LED indicator of A3CPU, A3NCPU, and A3HCPU by the display instructions are as follows:

- Numeral: 0 to 9
- Alphabet: A to Z (Capitals)
- Special symbol: <, >, =, *, /, ,, +, −

7.8.1 ASCII code print instructions (PR, PRC)

Processing Unit	Applicable CPU			
	A1N	A2N	A3N	A3H
——				

		Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level							
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
PR	Ⓢ								○	○	○	○	○											7		○		○	○
	Ⓓ		○																										
PRC	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○							○	○								
	Ⓓ		○																										



□ **Indicates the instruction symbol.**

PR, PRC

Setting data

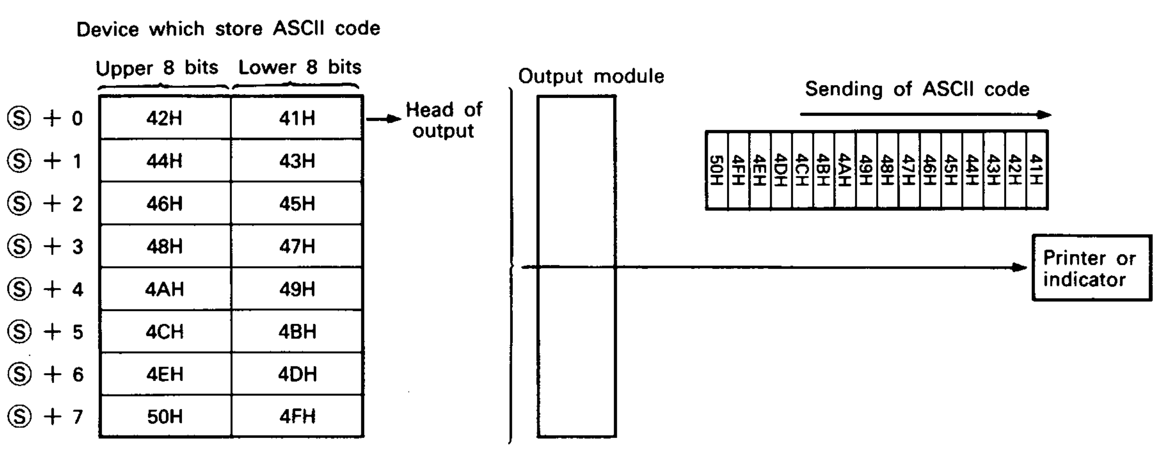
PR	Ⓢ	Head number of device which stores ASCII code
	Ⓓ	Head number of output unit which outputs ASCII code
PRC	Ⓢ	Device number which prints comment
	Ⓓ	Head number of output unit which outputs comment

Functions

PR

A1N, A2N, A3N, when M9049 turns ON

(1) Outputs an ASCII code of 16 characters stored in units of eight points beginning with the device specified at Ⓢ, to the output unit specified at Ⓓ. The number of points used for the output unit is ten points which start at the Y number specified at Ⓓ.



(2) The output signal from the output unit is sent at 30ms per character. Therefore, 480ms (= 16 × 30ms) is required until 16 characters are sent. However, since the control during sending is performed by the interrupt processing at intervals of 10ms, the sequence processing is performed continuously.
10 points beginning with the Y number specified in D are provided to the output unit during sequence processing, irrespective of I/O refresh after END.

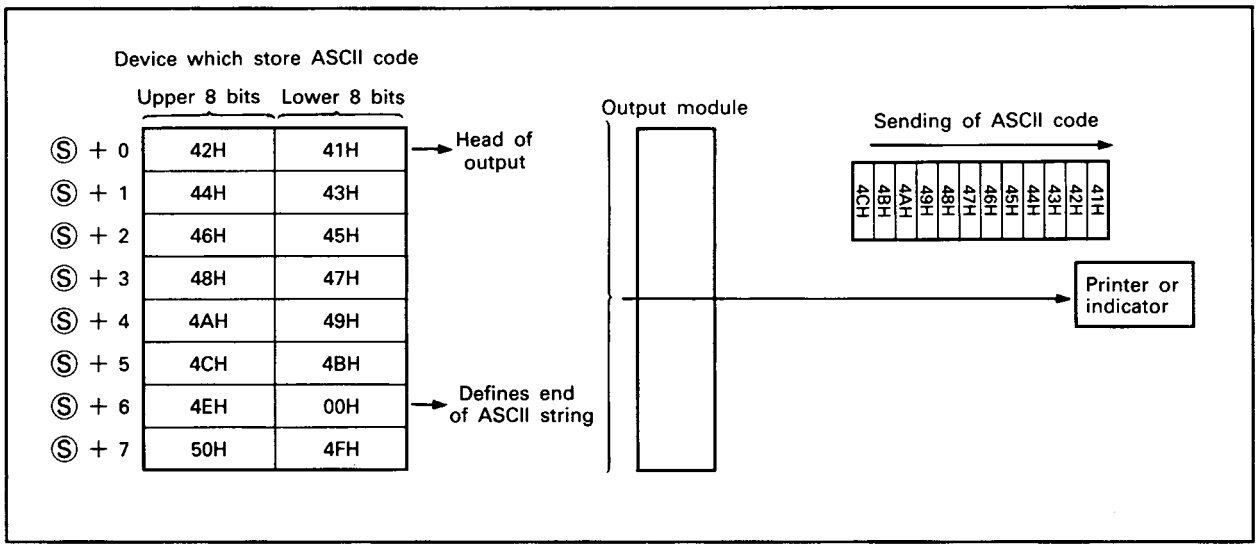
(3) In addition to the ASCII code, a strobe signal (10ms ON, 20ms OFF) is also output from the output unit.

(4) Until the completion of sending the ASCII code of 16 characters after the execution of PR instruction, the PR instruction execution flag is on.

(5) Multiple PR and PRC instructions can be used. In such a case, however, provide interlock by use of the PR instruction execution flag so that the instructions do not turn on at the same time.

A1N, A2N, A3N, A3HCPU, when M9049 turns OFF

(1) Outputs ASCII codes from the specified device, Ⓢ, to the specified output module, Ⓓ, until code 00H is given. The number of points used by the output module is 10, beginning with the Y number, Ⓓ.



(2) 480ms is required to transmit 16 codes as each code is transmitted 30ms by the output module (16 × 30ms = 480ms). The PR instruction performs processings during 10ms interrupts in order of data output, strobe signal on, strobe signal off. Any other instruction is executed between the processings.

(3) In addition to the ASCII code, a strobe signal (10ms ON, 20ms OFF) is also output from the output module.

(4) The PR instruction execution flag is switched on between the PR instruction executed and the specified number of ASCII codes transmitted.

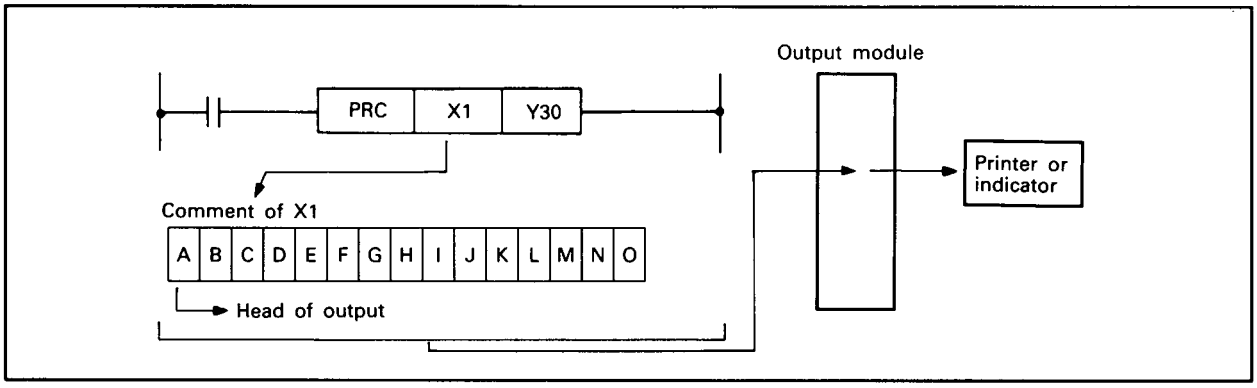
(5) Multiple PR and PRC instructions can be used. In such a case, however, provide interlock by use of the PR instruction execution flag so that the instructions do not turn on at the same time.

(6) Output data will change as it is not saved.

(7) An error will occur if there is no NUL (00H) code in the specified device.

PRC

(1) Outputs the comment (ASCII code) of the device specified at Ⓢ to the output unit specified at Ⓓ. The number of points used for the output module is eight points which start at the Y number specified at Ⓓ.



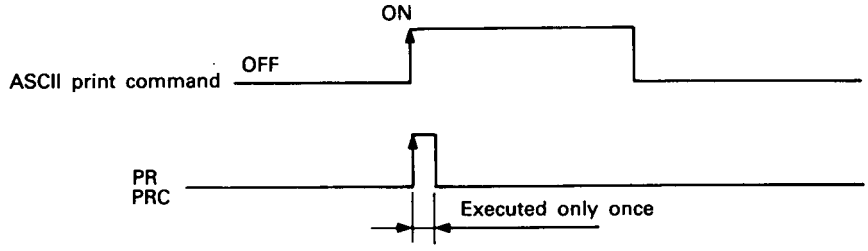
(2) 480ms is required to transmit 16 codes as each code is transmitted 30ms by the output module (16 × 30ms = 480ms). The PRC instruction performs processings during 10ms interrupts in order of data output, strobe signal on, strobe signal off. Any other instruction is executed between the processings.

(3) In addition to the ASCII code, a strobe signal (10ms ON, 20ms OFF) is also output from the output module.

(4) Until the completion of sending the ASCII code of 16 characters after the execution of PRC instruction, the PRC instruction execution flag is on.

(5) Multiple PR and PRC instructions can be used. In such a case, however, provide interlock by use of the PRC instruction execution flag so that the instructions do not turn on at the same time.

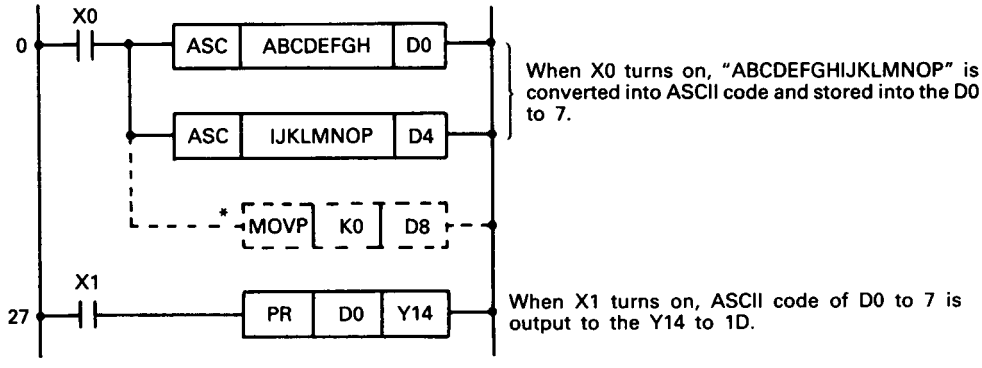
Execution conditions



Program Examples

PR

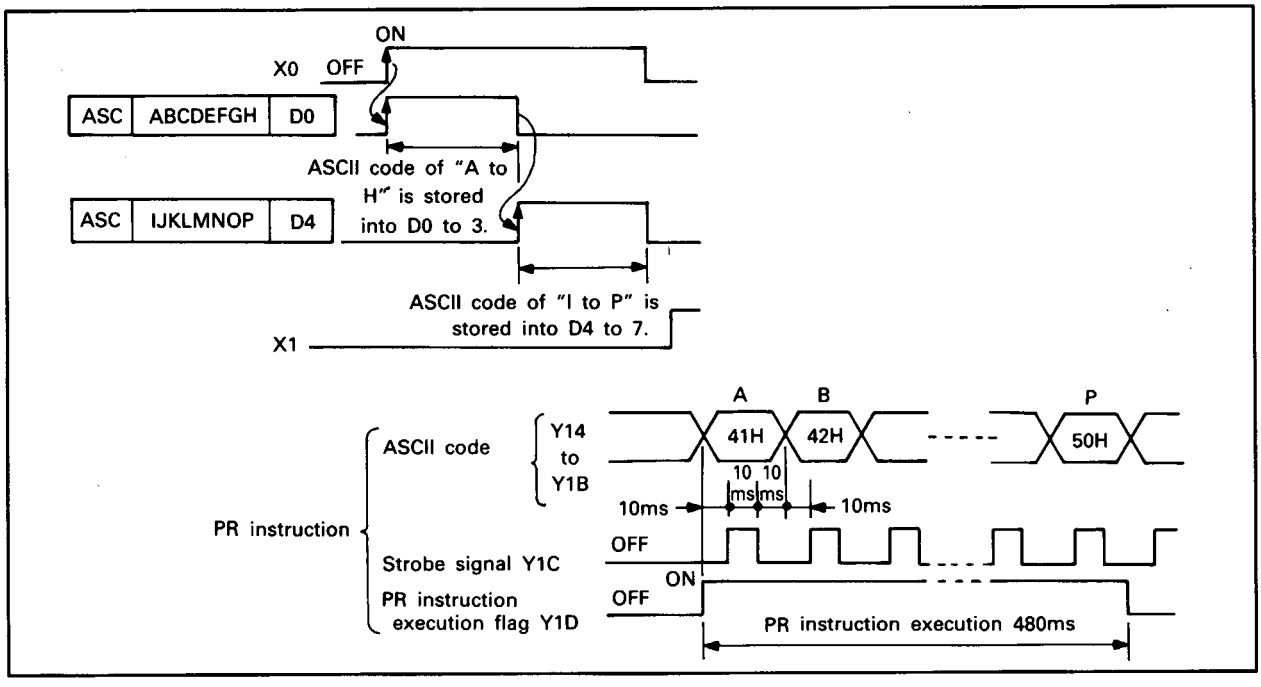
Program which converts "ABCDEFGHIJKLMNOP" into an ASCII code and stores the code into the D0 to 7 when X0 turns on, and outputs the ASCII code of D0 to 7 into the Y14 to 1D when X1 turns on.



Coding

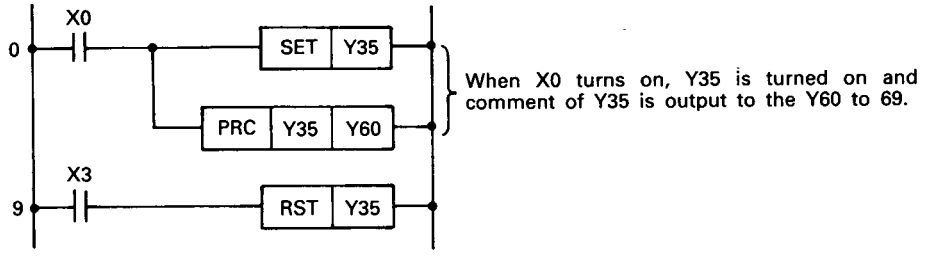
Step Number	Instruction	Device			
0	LD	X0			
1	ASC	ABCDEFGH	D0		
14	ASC	IJKLMNOP	D4		
27	LD	X1			
28	PR	D0	Y14		
35	END				

*: When the A1N, A2N, A3N or A3HCPU is used, 00H must be specified in D8 in this example as an error will result without the NUL (00H) code.



PRC

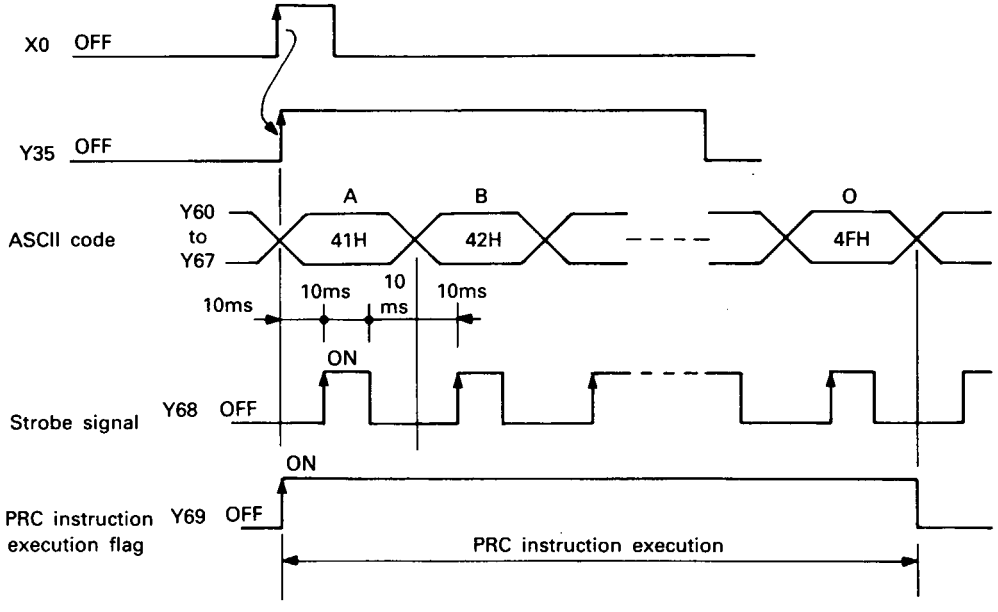
Program which turns on Y35, and at the same time, outputs the comment of Y35 to the Y60 to 69 when X0 turns on.



Coding

Step Number	Instruction	Device			
0	LD	X0			
1	SET	Y35			
2	PRC	Y35	Y60		
9	LD	X3			
10	RST	Y35			
11	END				

When comment of Y35 is A B C D E F G H I J K L M N O

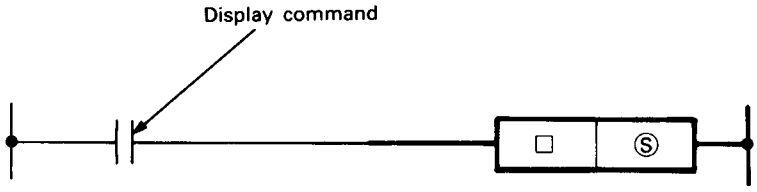


7

7.8.2 ASCII code comment display instructions (LED, LEDC)

Processing Unit	Applicable CPU			
	~~A1N~~	~~A2N~~	A3N	A3H

		Available Device																			Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
		Bit device							Word (16-bit) device									Constant		Pointer		Level					M9012	M9010	M9011
		X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N							
LED	Ⓢ								○	○	○	○	○											3		○		○	○
LEDC	Ⓢ	○	○	○	○	○	○	○	○	○	○	○	○							○	○								



□ Indicates the instruction symbol.

LED, LEDC

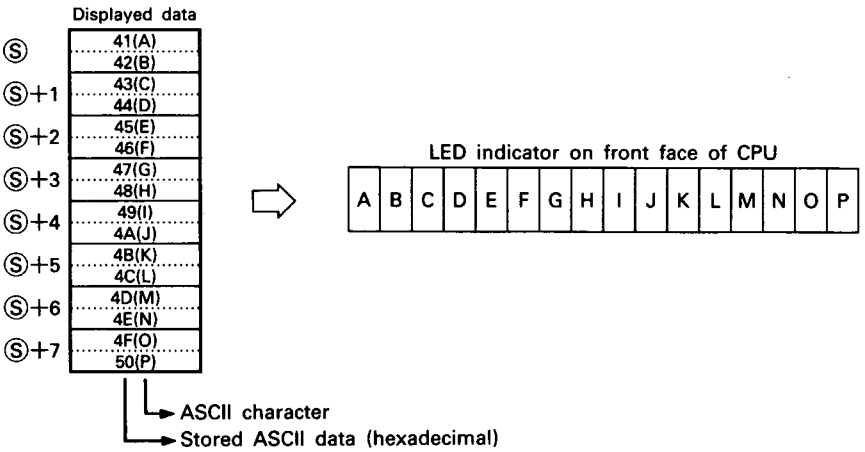
Setting data

Ⓢ	LED	Head number of device which stores displayed data
	LEDC	Device number of which comment will be displayed

Functions

LED

(1) Displays the ASCII data (16 characters) stored at eight points, which begin with the device specified at Ⓢ, at the LED indicator on the front face of CPU.



(2) When the ASCII data is not stored at the eight points which begin with the device specified at Ⓢ,

1) T, C, D, W: Blank
2) R: What will be displayed is unknown.
(Blank when the file register (R) has been cleared.)

(3) For ASCII characters which can be displayed, refer to the Appendix.

(4) For the conversion of alphanumeric characters into ASCII data in a sequence program, use the ASC instruction.

IB (NA) 66147-A

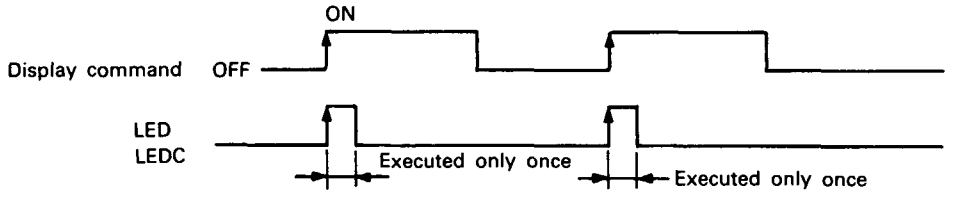
LEDC

(1) Displays the comment (15 characters) of device specified at Ⓢ at the LED indicator on the front of CPU.

(2) When the device specified at Ⓢ is not annotated with a comment, the action of LEDC is as described below:

Specification of Ⓢ		Operation of LED
Inside comment range specification	With comment	Comment of device is displayed at LED indicator
	Without comment	Display of LED indicator is cleared.
Outside comment range specification		No Processing (Display of LED indicator does not change.)

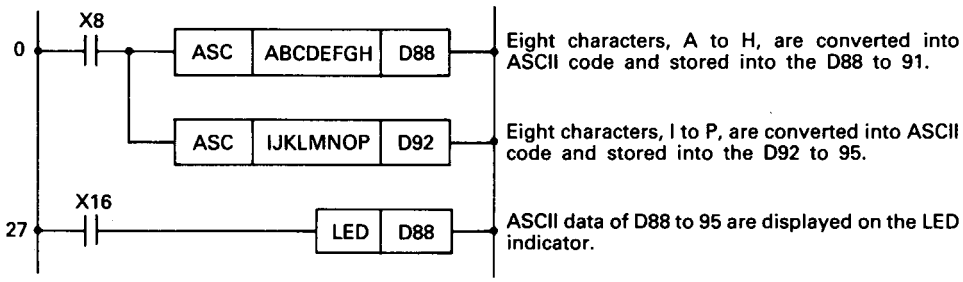
Execution Conditions



Program Examples

LED

Program which converts "ABCDEFGHIJKLMNOP" into ASCII code and stores it to the D88 to 95 when X8 turns on, and displays the ASCII data of D88 to 95 at the LED indicator on the front face of CPU when X16 turns on.

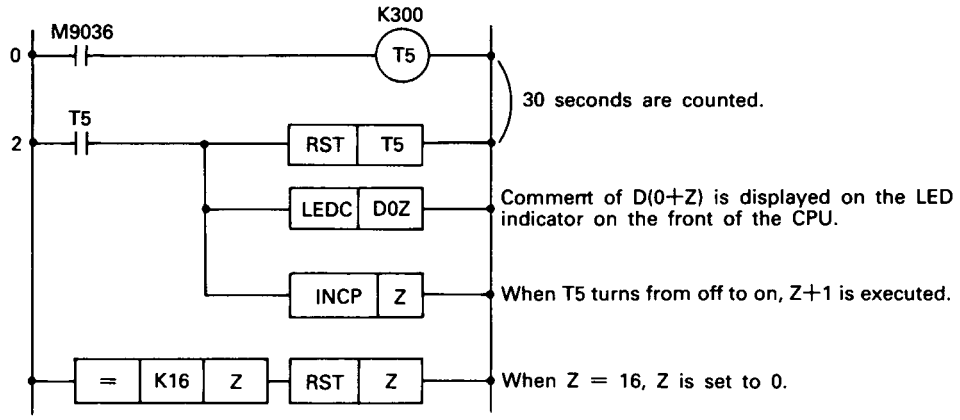


Coding

Step Number	Instruction	Device			
0	LD	X8			
1	ASC	ABCDEFGH	D88		
14	ASC	IJKLMNOP	D92		
27	LD	X16			
28	LED	D88			
31	END				

LEDC

Program which displays the comment of D0 to D15 at intervals of 30 seconds.



Coding

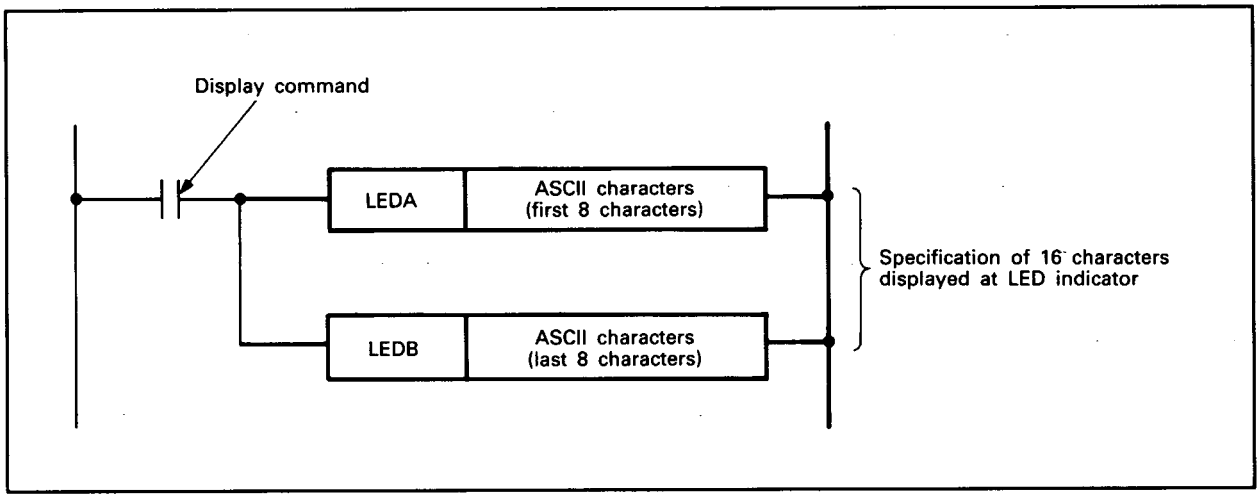
Step Number	Instruction	Device			
0	LD	M9036			
1	OUT	T5	K300		
2	LD	T5			
3	RST	T5			
6	LEDC	D0Z			
9	INCP	Z			
12	LD=	K16	Z		
17	RST	Z			
20	END				

MEMO

7.8.3 Character display instructions (LEDA, LEDB)

Processing Unit	Applicable CPU			
	~~A1N~~	~~A2N~~	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						13					



Functions

(1) Displays the ASCII characters specified by LEDA and LEDB at the LED indicator on the CPU front.

(2) The displays of LEDA and LEDB are as shown below.

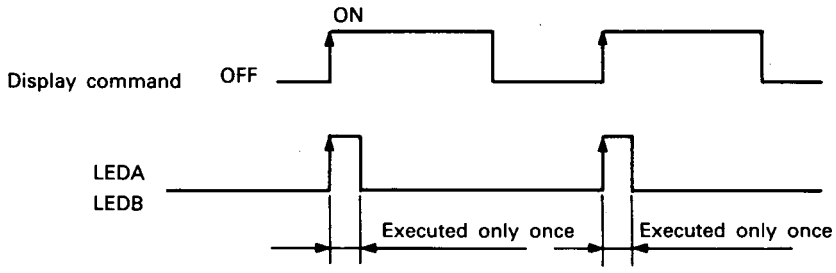
LED indicator at CPU front (16 characters)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P

LEDA ⇩ Specification of first half 8 characters

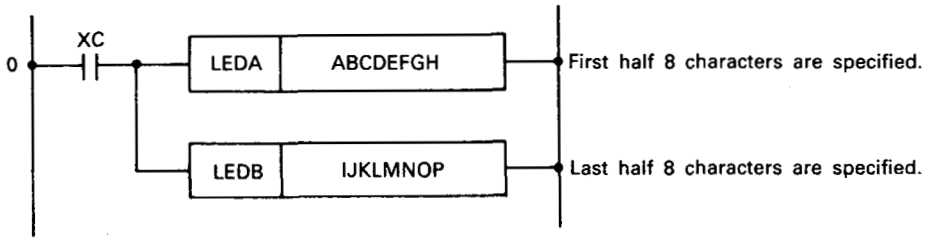
LEDB ⇩ Specification of last half 8 characters

Execution Conditions



Program Examples

Program which displays "ABCDEFGHIJKLMNOP" at the LED indicator on the CPU front when XC turns on.



Coding

Step Number	Instruction	Device			
0	LD	XC			
1	LEDA	ABCDEFGH			
14	LEDB	IJKLMNOP			
27	END				

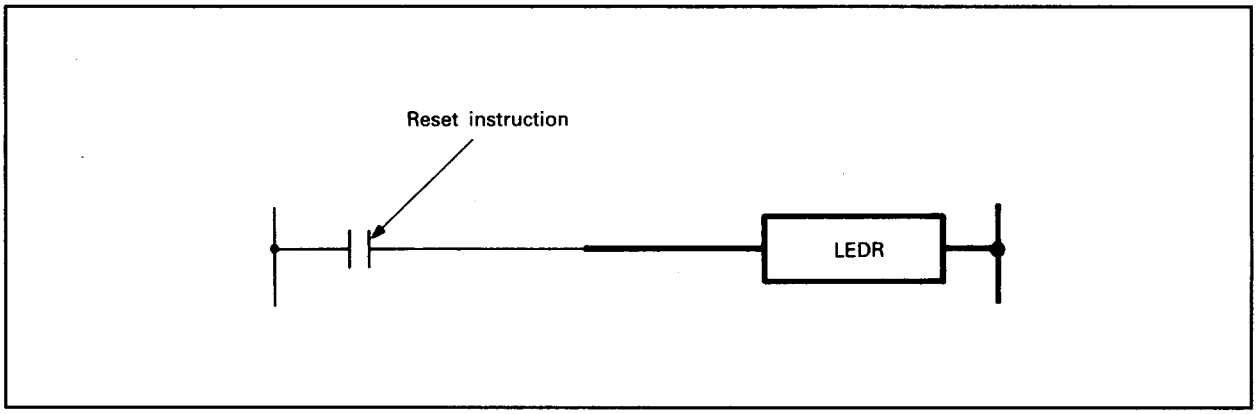
REMARKS

The second eight of the 16 characters displayed by the LED instruction will disappear if the first eight are rewritten by the LEDA instruction.
The first eight characters will disappear if the second eight are rewritten by the LED instruction.

7.8.4 Annunciator reset instruction (LEDR)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					

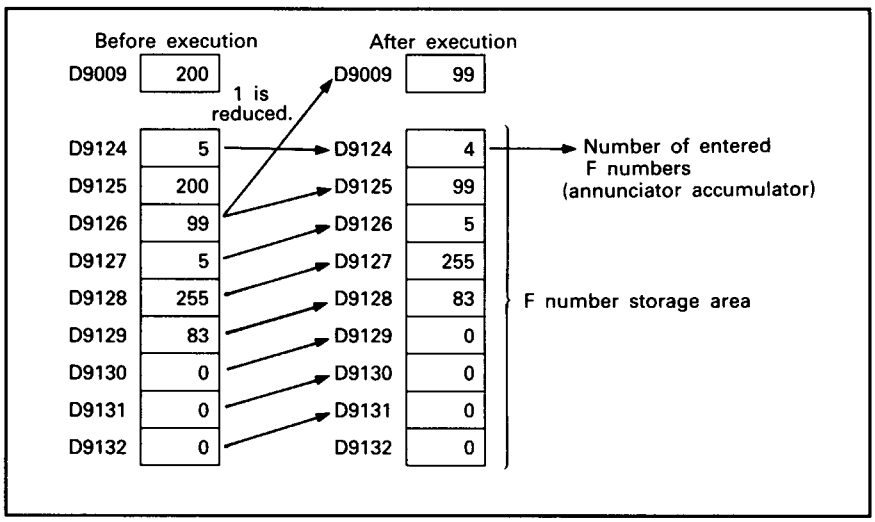


Functions

A1NCPU and A2NCPU

Performs the following actions:

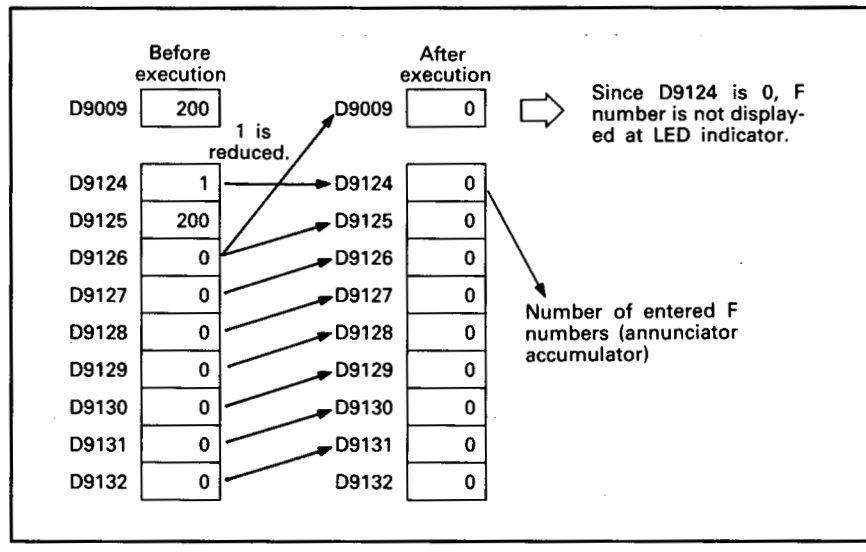
1) Flickers and then turns off the "ERROR" LED.
2) Resets the annunciator (F) stored in D9009.
3) Resets D9009 and 9125 and shifts the F numbers of D9126 to 9131 to be processed.
4) Transfers the F number, which has been newly stored in D9125, to D9009.
5) Reduces 1 from the data of D9124. However, when D9124 is 0, the data remains 0.



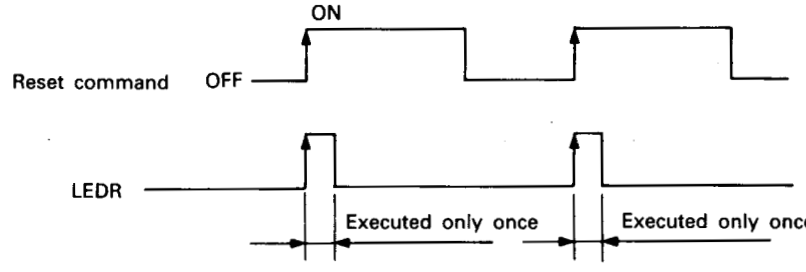
A3NCPU and A3HCPU

Performs the following actions:

1) Resets the F number displayed at the CPU front.
2) Resets the annunciator (F) stored in D9009.
3) Resets D9009 and 9125 and shifts the F numbers of D9126 to 9131 to be processed.
4) Transfers the F number, which has been newly stored in D9125, to D9009.
5) Reduces 1 from the data of D9124. However, when D9124 is 0, the data remains 0.
6) Displays the F number stored in D9009 at the LED indicator. (When D9124 is 0, the F number is not displayed.)



Execution Conditions



MEMO

7.9 Other Instructions

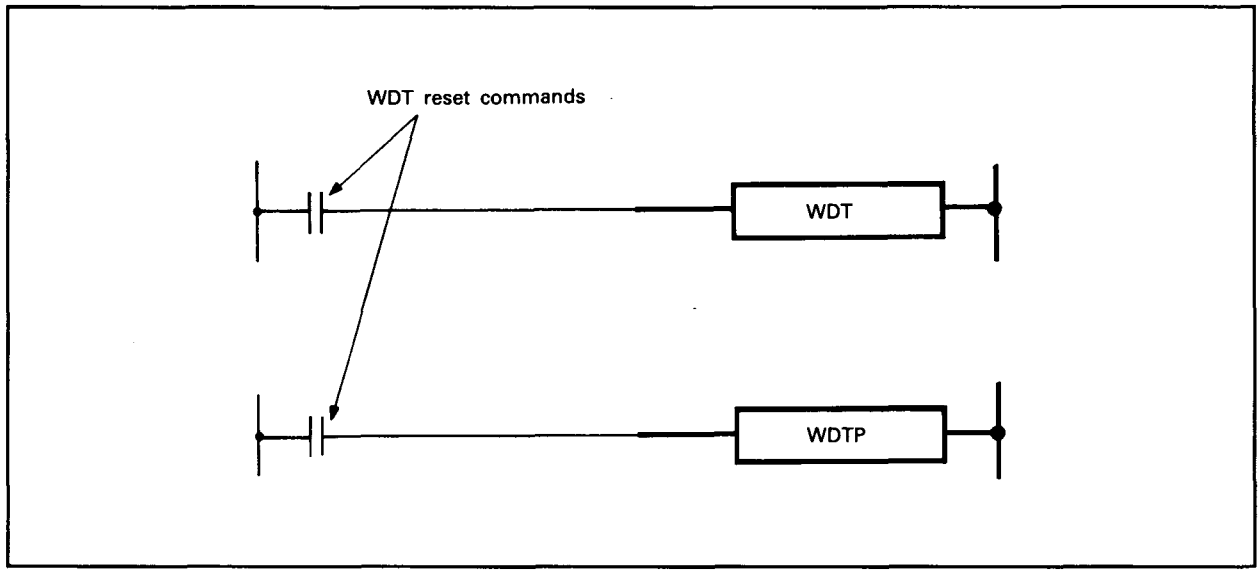
Instructions which perform operations such as the reset of WDT, the failure check, and the set and reset of carry flag.

Classification		Instruction Symbol	Ref. Page
WDT reset		WDT	7-87 to 7-88
Status latch	Set	SLT	7-89 to 7-90
	Reset	SLTR	7-89 to 7-90
Sampling trace	Set	STRA	7-91 to 7-92
	Reset	STRAR	7-91 to 7-92
Carry	Set	STC	7-93 to 7-94
	Reset	CLC	7-93 to 7-94
Timing clock		DUTY	7-95 to 7-96

7.9.1 WDT reset (WDT, WDTP)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					

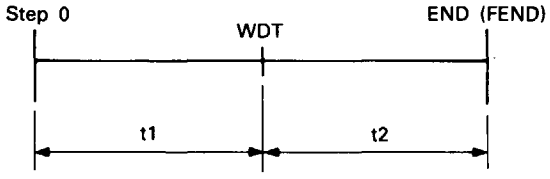


Functions

(1) Resets the watch dog timer in a sequence program.

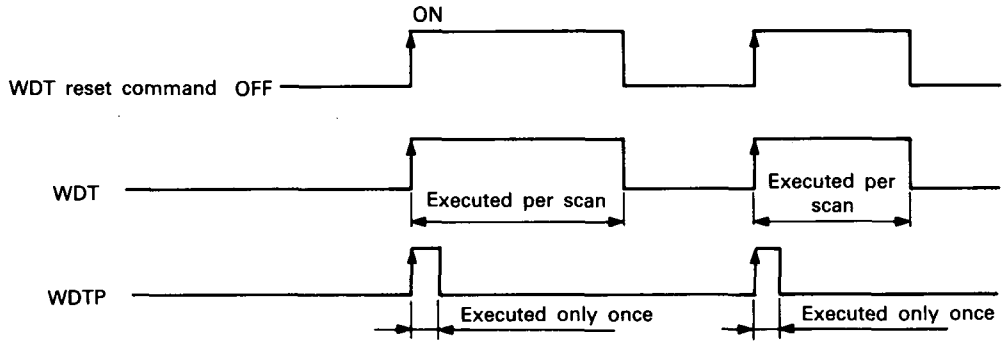
(2) Used when the period of time from step 0 to END (FEND) in the sequence program exceeds the set value of watch dog timer depending on conditions. If the scan time exceeds the set value of watch dog timer at every scan, change the set value of watch dog timer by the parameter setting of peripheral equipment (A6GPP, A6PHP, A6HGP, A7PU).

(3) Set the set value of the watch dog timer so that "t1" from step 0 to WDT instruction and "t2" from the WDT to END (FEND) instruction do not exceed the set value. (See the diagram below.)



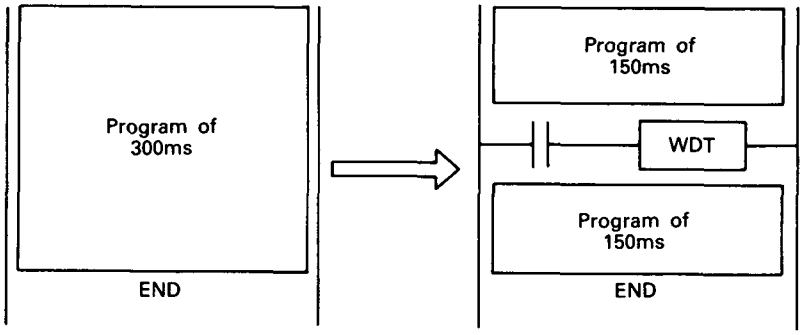
(4) The WDT instruction can be used two or more times during one scan. However, care should be exercised because, if error occurs, the outputs cannot be turned off immediately.

Execution Conditions



Program Example

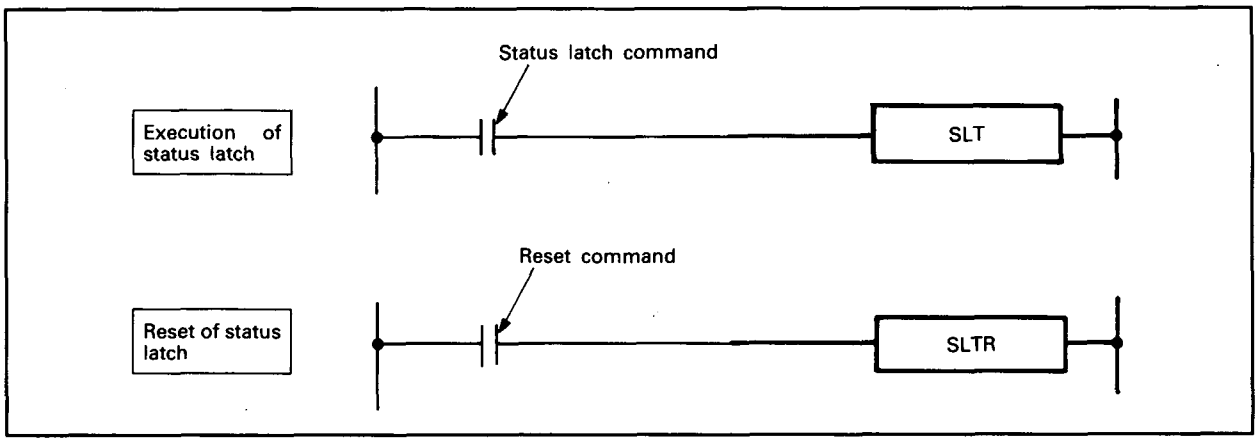
Program used when the setting of watch dog timer is 200ms and the period of time from 0 to END (FEND) instruction is 300ms depending on the execution conditions of program.



7.9.2 Status latch set, reset (SLT, SLTR)

Processing Unit	Applicable CPU			
	~~A1N~~	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

SLT

(1) When executed, the SLT instruction stores the contents of data memories and file registers set by the parameter setting of peripheral unit A6GPP, A6PHP, A6HGP into the memory for status latch in the user memory area.

(2) Status latch is allowed for the following devices.

Data memory: ON/OFF displays of X, Y, M, B, and F
Present values of T and C
Contents of D, W, A0, A1, Z and V
Contents of file registers

(3) When the SLT instruction is executed only once.

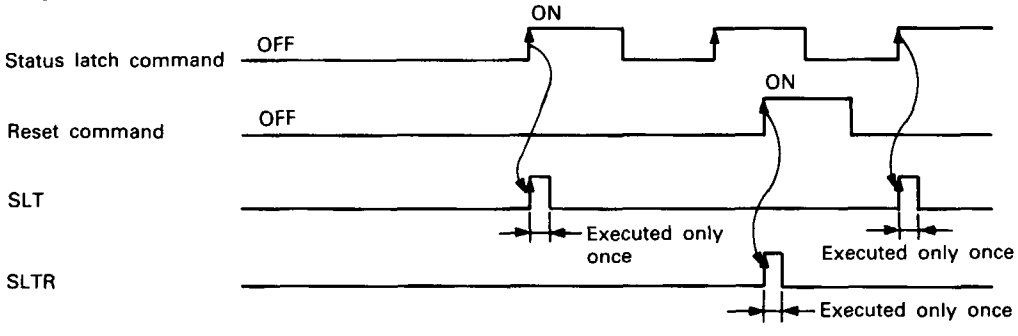
(4) The result of status latch can be monitored by the A6GPP, A6PHP, A6HGP.

SLTR

(1) A reset instruction of SLT instruction.

(2) By executing the SLTR instruction, the SLT instruction is enabled again.

Execution Conditions



POINT

When the status latch (SLT) instruction is executed, the scan time of programmable controller CPU increases as shown in the following table.

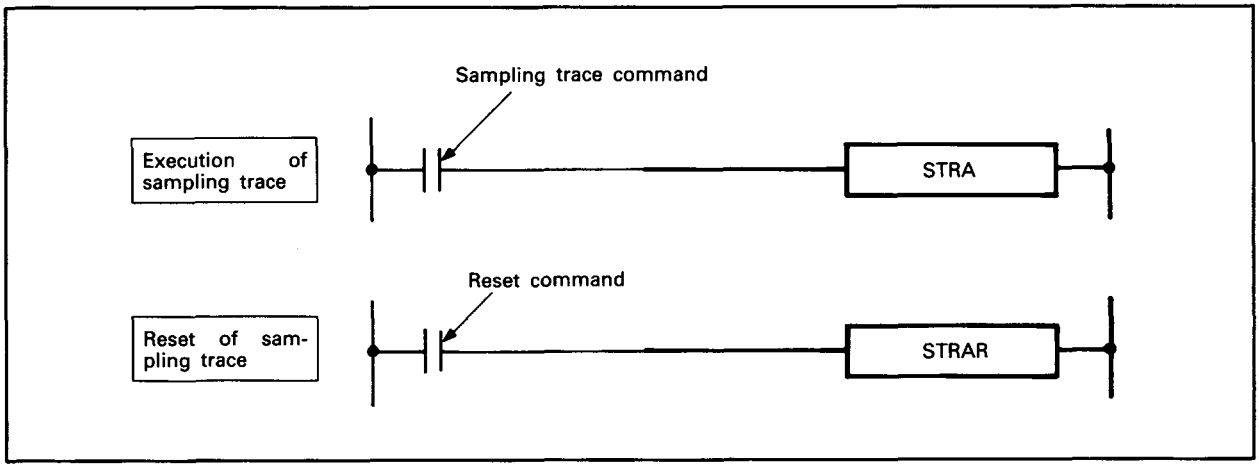
	Latch of Only Device Memory	Latch of Both Device Memory and File Register
A2NCPU	8.5ms	25ms
A3NCPU	8.5ms	37ms
A3HCPU	4.1ms	10.4ms

Set the watch dog timer of programmable controller CPU after considering the above increase in scan time.

7.9.3 Sampling trace set, reset (STRA, STRAR)

Processing Unit	Applicable CPU			
	~~A1N~~	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level						
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

STRA

(1) When M9047 is switched on, the sampling trace data specified by the peripheral device is stored to the dedicated memory area the specified number of times. After the specified number of times is reached, the data sampled is latched and the sampling trace is stopped.
(If M9047 turns off during the sampling, the sampling is stopped.)

(2) Sampling trace data are as follows:

X, Y, M, B, F, T/C (coil, contact): Maximum of eight contacts
T, C, D, W, R, A0, A1, Z, V: Maximum of three points

(3) Upon completion of the sampling trace after the execution of STRA instruction, M9043 turns on.

(4) The STRA instruction is executed only once.

(5) The sampling trace result can be monitored by the peripheral device.

(6) The STRA and STRAR instructions cannot be executed during ROM operation.

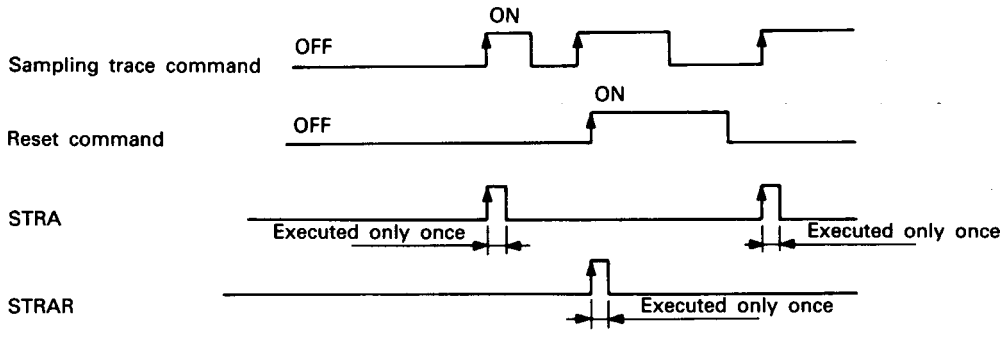
STRAR

(1) Reset instruction for the STRA instruction.

(2) By executing the STRAR instruction, the STRA instruction is enabled again.

(3) Turns off M9043.

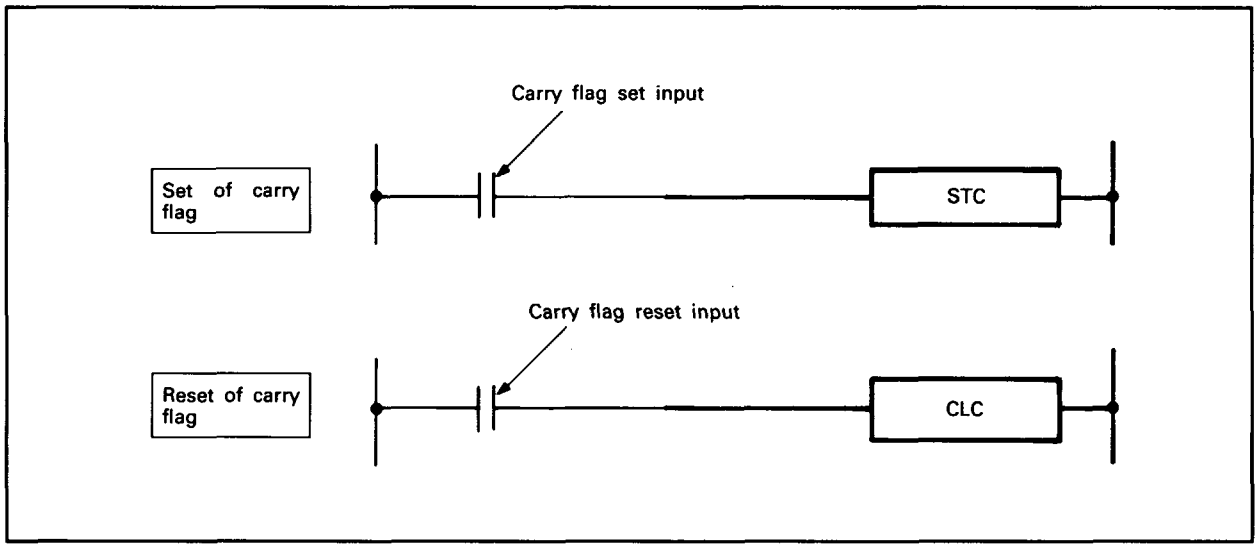
Execution Conditions



7.9.4 Carry flag set, reset (STC, CLC)

Processing Unit	Applicable CPU			
———	A1N	A2N	A3N	A3H

Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
Bit device							Word (16-bit) device									Constant		Pointer		Level							
X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
																						1					



Functions

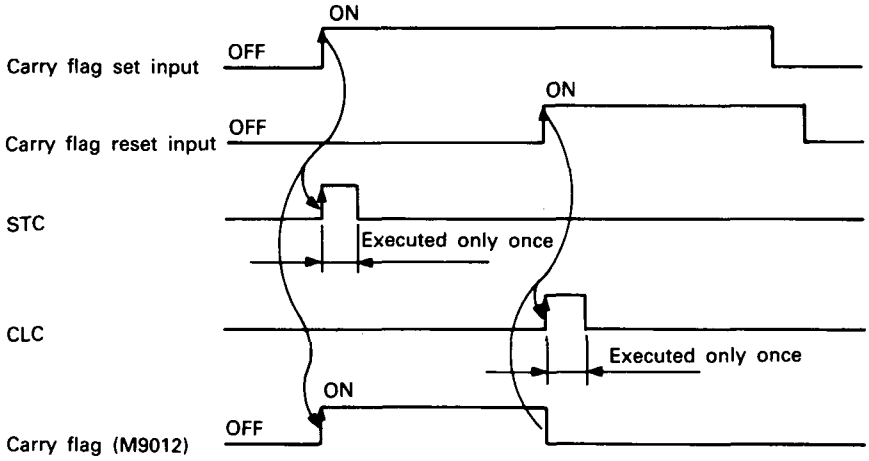
STC

(1) Sets (turns on) the carry flag contact (M9012).

CLC

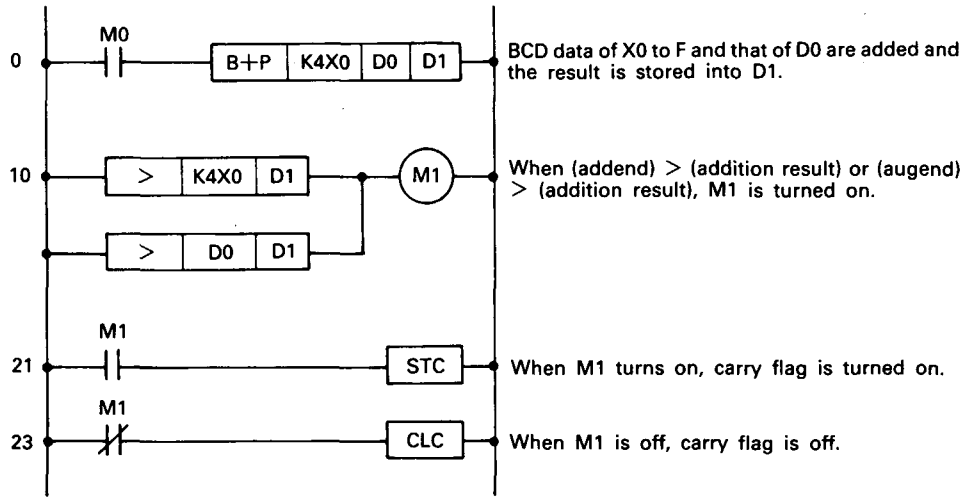
(1) Resets (turns off) the carry flag contact (M9012).

Execution Conditions



Program Example

Program which performs addition of the BCD data of X0 to F and the BCD data of D0 when M0 turns on, and turns on the carry flag (M9012) when the result is more than 9999, and turns off the carry flag when the result is 9999 or less.



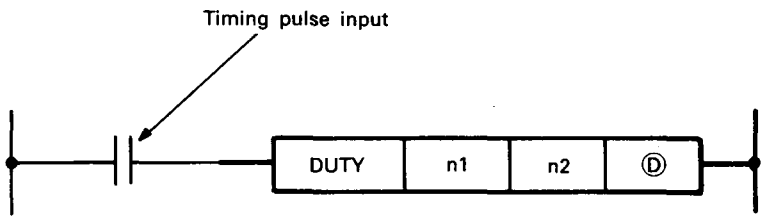
Coding

Step Number	Instruction	Device			
0	LD	M0			
1	B+P	K4X0	D0	D1	
10	LD>	K4X0	D1		
15	OR>	D0	D1		
20	OUT	M1			
21	LD	M1			
22	STC				
23	LDI	M1			
24	CLC				
25	END				

7.9.5 Pulse regeneration instruction (DUTY)

Processing Unit	Applicable CPU			
1 bit	A1N	A2N	A3N	A3H

	Available Device																				Digit specification	Number of steps	Subset	Index	Carry flag	Error flag	
	Bit device							Word (16-bit) device									Constant		Pointer		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
n1																	○	○										
n2																	○	○					7				○	○
Ⓓ			○																									



Setting data

n1	Number of scans during which timing pulse is on
n2	Number of scans during which timing pulse is off
Ⓓ	Timing clocks for user (M9020 to 4)

Functions

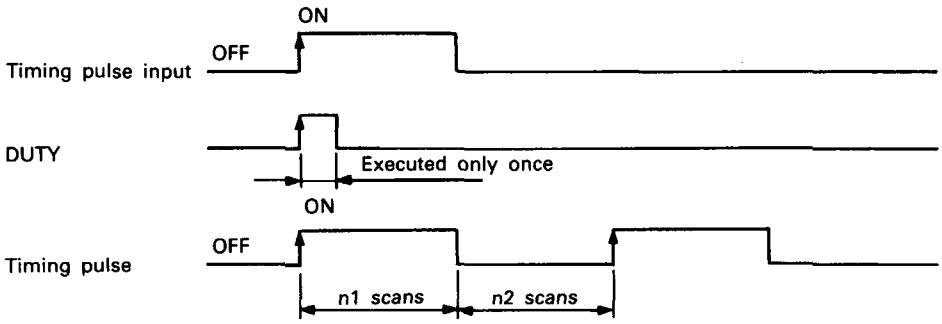
(1) Sets the timing clock for user (M9020 to 9024) specified at Ⓓ to ON at the scan count specified at "n1" and to OFF at the scan count specified at "n2".

(2) At the initial status (when the timing pulse input is off), the timing pulse is off.

(3) When "n1" and "n2" are set to 0, the timing pulse is as described below:

"n1" = 0: The timing pulse remains off.
"n1" > 0, "n2" = 0: The timing pulse remains on.

Execution Conditions

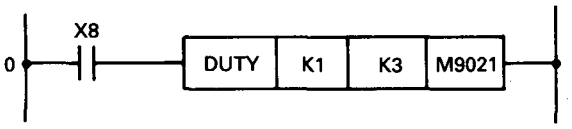


Operation Error

In the following case, operation error occurs and the error flag turns on.

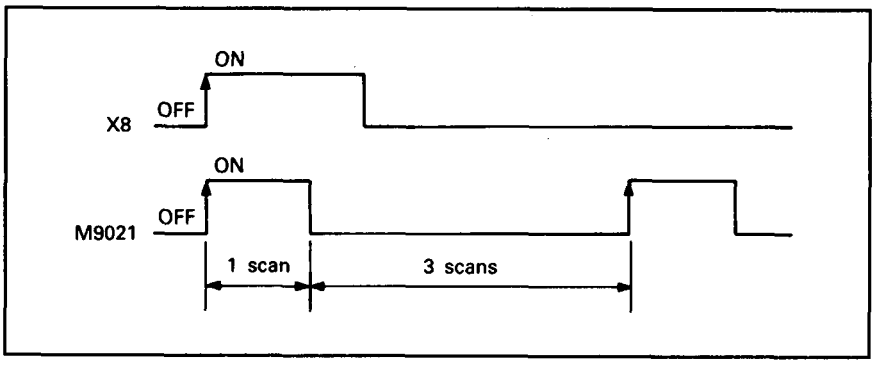
○ The setting of D is other than M9020 to 9024.

Program Example



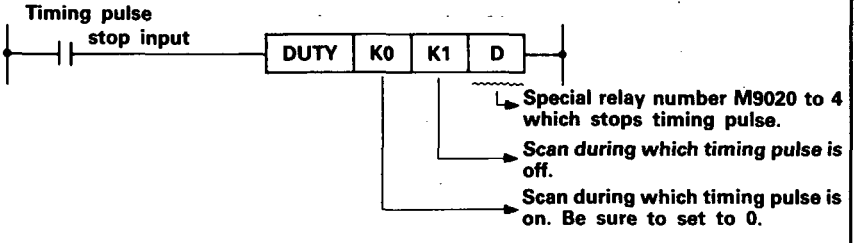
Coding

Step Number	Instruction	Device			
0	LD	X8			
1	DUTY	K1	K3	M9021	
8	END				



POINT

Even if the timing pulse input turns off, the timing pulse by the DUTY instruction does not turn off. Therefore, to stop the timing pulse, execute the circuit as shown below.



8. MICROCOMPUTER PROGRAM

8.1 Specifications of Microcomputer Mode

Item	Specifications	
Module	A1N, A2N, A3NCPU	A3HCPU
CPU	8086 (10MHz)	80286 (8MHz)
Microcomputer program area	0 to 10K bytes·····A1NCPU 0 to 26K bytes·····A2NCPU 0 to 58K bytes (Main microcomputer area) 0 to 58K bytes (Sub microcomputer area) } ·····A3NCPU	0 to 58K bytes (Main microcomputer area) 0 to 58K bytes (Sub microcomputer area) } ·····A3HCPU
	Specify the microcomputer program area in multiples of 2K bytes. The relation between the main (sub) program, sequence program, and microcomputer program capacities is as indicated below: (Main (sub) program memory capacity) = (sequence program memory capacity) + (microcomputer program memory capacity)	
Work area	A100H to A1FFH (256 bytes)	
Stack area	User area: 128 bytes (No setting required by the user)	
Instructions which cannot be used	INT, INTO, IRET, IN, OUT, HLT, WAIT, LOCK, ESC	INT, INTO, IRET, IN, OUT, HLT, WAIT, LOCK, ESC, CLI, STI

Table 8.1 Specifications of Microcomputer Mode

The A series permits the combined processing of sequence mode and microcomputer mode which calls and runs the user-created microcomputer program during the run of sequence program (subsequence program) and returns the execution to the sequence program (subsequence program) again. (Design and debug the microcomputer program on user side.)

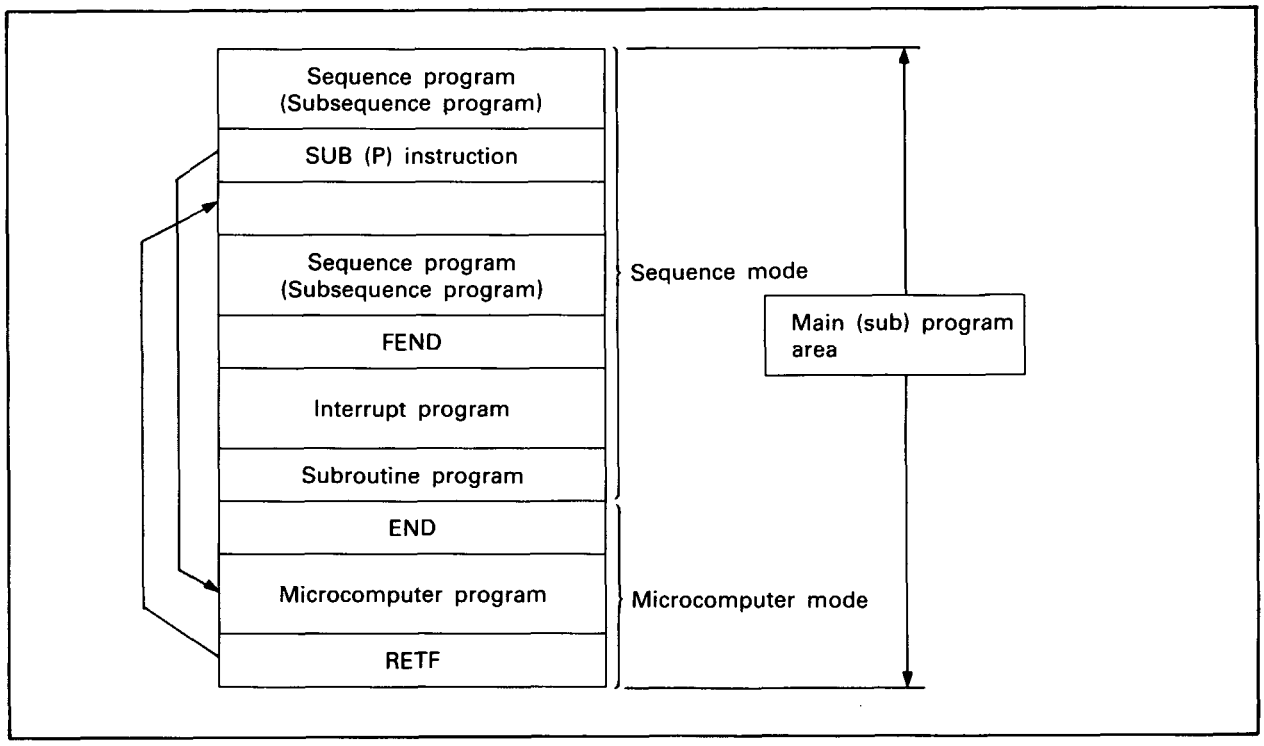


Fig. 8.1 Call of Microcomputer Program

8.2 Using Utility Program

Various types of control and operation (e.g. PID control, function operation, code conversion) can be executed by calling the utility program from the microcomputer program area.

(1) Utility program entry procedure

Combine together the utility program with the user program in the following procedure:

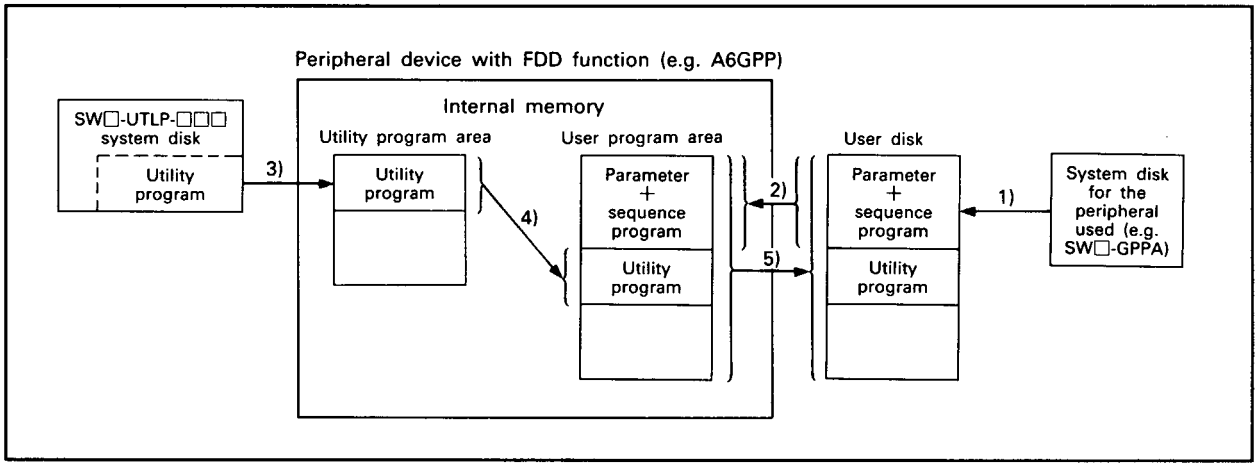


Fig. 8.2 Entering the Utility Program

1) Specify parameters and write the sequence program using the peripheral device.

2) Load the SW□-UTLP-□□□ system disk into the peripheral device and read the parameters and sequence program from the user disk to the user program area.

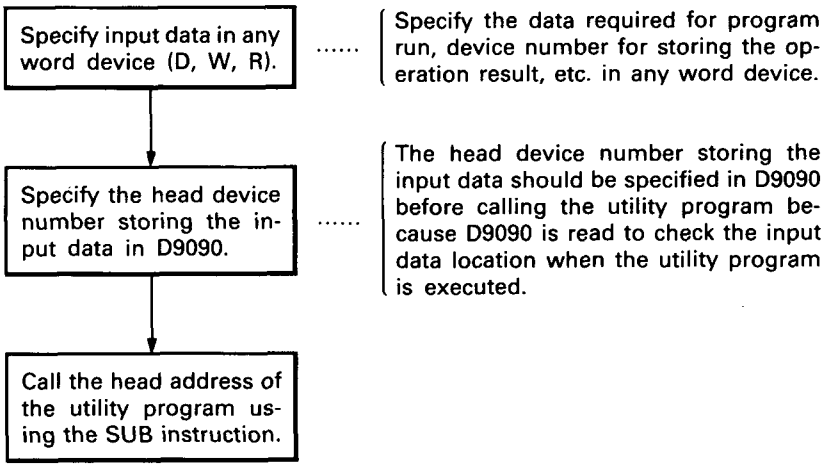
3) Read the utility program from the system disk to the utility program area.

4) Combine together the sequence program and utility program in the user user program area.

5) Write the combined program onto user disk.

(2) Calling the utility program

Call the utility program from the sequence program as described below:



For further information, see the corresponding utility program operating manual.

8.3 Using User-Written Microcomputer Program

(1) Calling method of microcomputer program

The microcomputer program is called by the execution of SUB instruction in the sequence program.
The format of the SUB instruction is as shown below.

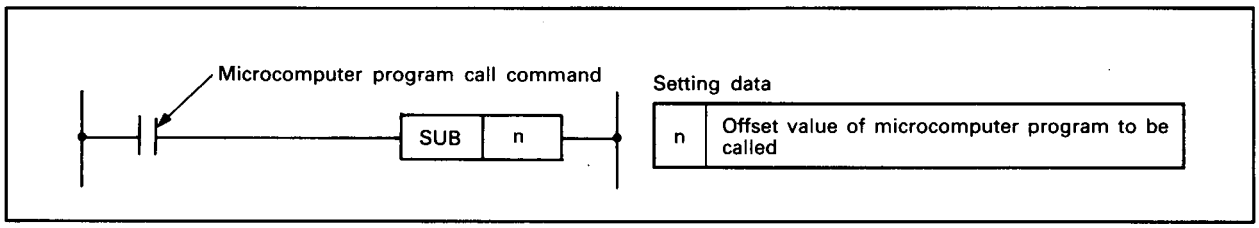
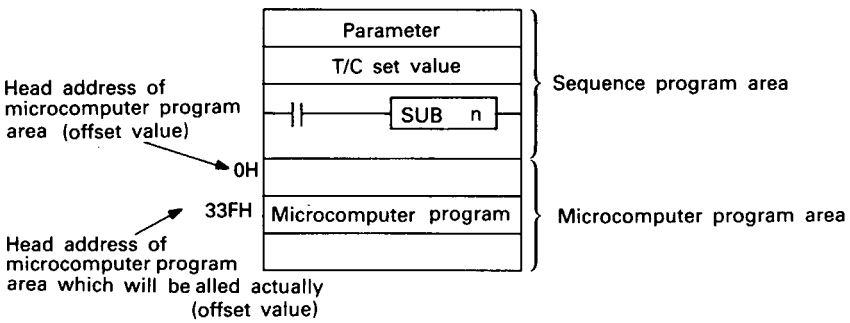


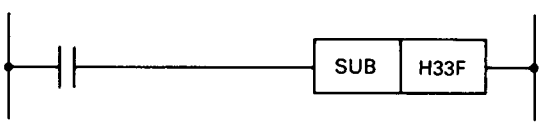
Fig. 8.3 Format of SUB Instruction

Example:

In the following memory map, the specification of "n" is as shown below.



In the SUB instruction, specify as shown below.



By changing the offset value specified at "n", multiple microcomputer programs can also be called.

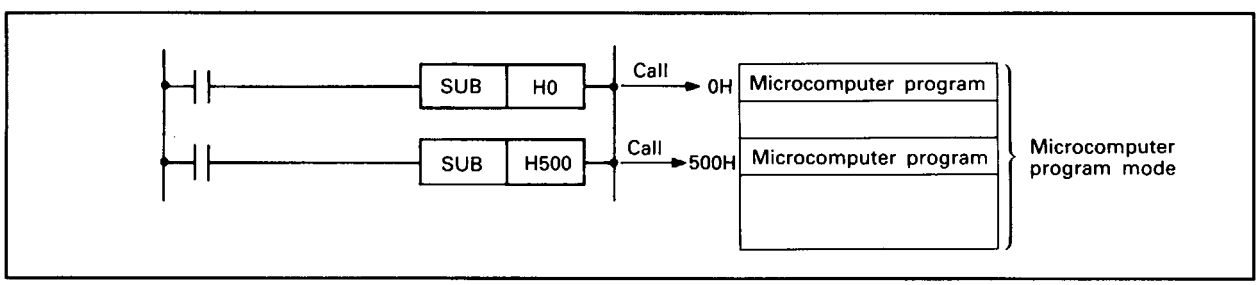


Fig. 8.4 Calling Method for Multiple Microcomputer Programs

8.3.1 Memory map

The microcomputer program may be used in the following areas.

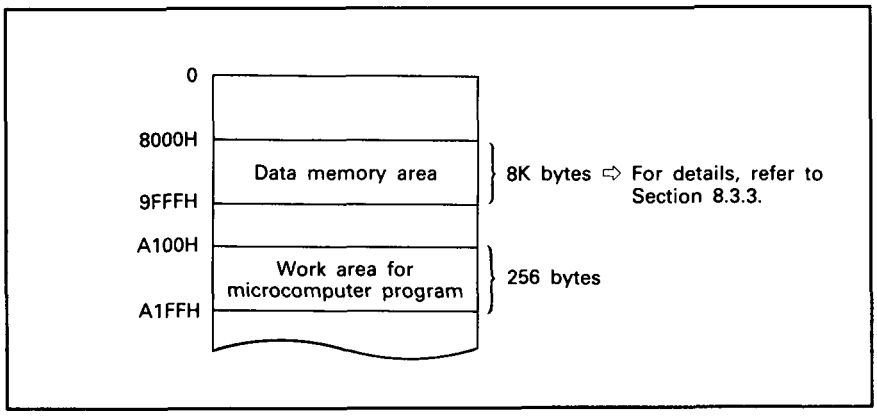


Fig. 8.5 Data Memory and Work Areas

8.3.2 Data memory area address configuration

One address of the data memory area consists of 16 bits which are further divided into the odd and even areas (i bits respectively).

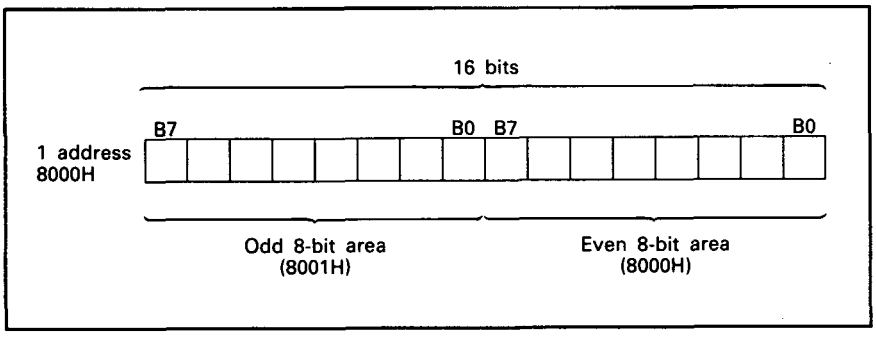


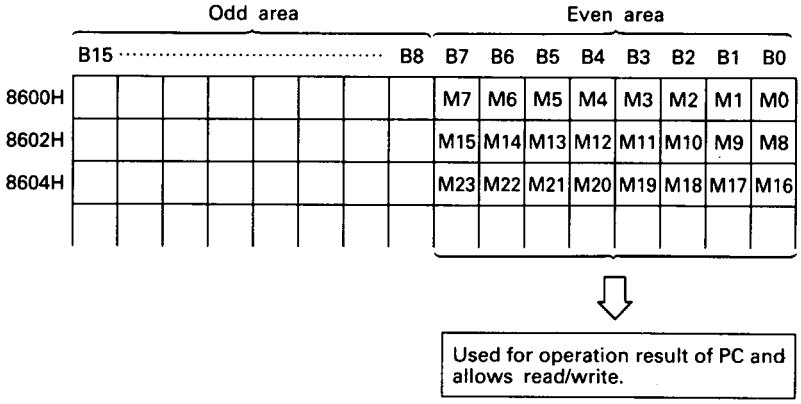
Fig. 8.6 Configuration of 1 Address (16 bits)

8.3.3 Configuration of data memory area

The data memory area (8000_H to $9FFF_H$) stores device data. The memory area of each device and its configuration are as indicated below.

Device	CPU Type	Address	Configuration
Input (X)	A1N	8000_H to $803F_H$: X0 to FF	Odd address: B15 B14 B13 B12 B11 B10 B9 B8 / Even address: B7 B6 B5 B4 B3 B2 B1 B0 8000_H XIM7 XIM6 XIM5 XIM4 XIM3 XIM2 XIM1 XIM0 X7 X6 X5 X4 X3 X2 X1 X0 8002_H XIMF XIME XIMD XIMC XIMB XIMA XIM9 XIM8 XF XE XD XC XB XA X9 X8 8004_H XIM17 XIM16 XIM15 XIM14 XIM13 XIM12 XIM11 XIM10 X17 X16 X15 X14 X13 X12 X11 X10 • Used for storing ON/OFF data from remote station and allows read/write. • Stored data area as follows: 0: OFF 1: ON • Used for storing ON/OFF data from input unit and allows only read. • Stored data area as follows: 0: ON 1: OFF Obtain actual input by the following expression: Input (X) = (XIM) ∨ (X)
	A2N	8000_H to $807F_H$: X0 to 1FF	
	A3N	8000_H to $81FF_H$: X0 to 7FF	
Output (Y)	A1N	8200_H to $823F_H$: Y0 to FF	Odd address: B15 ····· B8 / Even address: B7 B6 B5 B4 B3 B2 B1 B0 8200_H Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0 8202_H YF YE YD YC YB YA Y9 Y8 8204_H Y17 Y16 Y15 Y14 Y13 Y12 Y11 Y10 • Used for storing operation result of PC and allows read/write. • Stored data are as follows: 0: OFF 1: ON Read/write from/to output memory are performed as shown below: Write, Read, Output memory, Output unit, Direct mode, Refresh mode Output refresh after END instruction is executed
	A2N	8000_H to $827F_H$: Y0 to 1FF	
	A3N	8200_H to $83FF_H$: Y0 to 7FF	

Device	CPU Type	Address	Configuration
Internal relay (M) Latch relay (L) Step relay (S)	A1N A2N A3N	8400H to 85FFH — M/L/S 0 to 2047	○ All devices consist of one bit and store ON/OFF data of device by use of eight bits at even addresses. ○ ON/OFF of each device are as shown below: 0: OFF 1: ON Example M0 to 23 are as shown below:
Link relay (B)	A1N A2N A3N	8600H to 86FFH — B0 to 3FF	
Annunciator (F)	A1N A2N A3N	8700H to 873FH — F0 to 255	
Special relay (M)	A1N A2N A3N	8740H to 877FH — M9000 to 9255	
Contact of timer (T)	A1N A2N A3N	8780H to 87BFH — T0 to 255	
Contact of counter (C)	A1N A2N A3N	87C0H to 87FFH — C0 to 255	
Coil of timer (T)	A1N A2N A3N	9C00H to 9C3FH — T0 to 255	
Coil of counter (C)	A1N A2N A3N	9C40H to 9C7FH — C0 to 255	



Device	CPU Type	Address	Configuration
Data register (D)	A1N A2N A3N	8800H to 8FFFH — D0 to 1023	All devices consist of two bytes (16 bits). Example The configuration of D0 is as shown below: B7 B0 / 8800H (L) / 8801H (H) / B15 B8
Link register (W)		9000H to 97FFH — W0 to 3FF	
Present value of timer (T)		9800H to 99FFH — T0 to 255	
Present value of counter (C)		9A00H to 9BFFH — C0 to 255	
Special register (D)		9D00H to 9EFFH — D9000 to 9255	
Accumulator (A0, 1)		9FF8H — A0; 9FFAH — A1	
Index (Z, V)		9FFCH — Z; 9FFEH — V	

Device	CPU Type	Address	Configuration
Input (X)	A3H	8000H to 80FFH: X0 to 7FF	Odd address / Even address B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 8000H XF XE XD XC XB XA X9 X8 X7 X6 X5 X4 X3 X2 X1 X0 8002H X1F X1E X1D X1C X1B X1A X19 X18 X17 X16 X15 X14 X13 X12 X11 X10 8004H X2F X2E X2D X2C X2B X2A X29 X28 X27 X26 X25 X24 X23 X22 X21 X20 • Stores ON/OFF data from an input unit, read only. • 0 indicates ON and 1 OFF.
Output (Y)		8000H to 80FFH: Y0 to 7FF	Odd address / Even address B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 8200H XF XE XD XC XB XA X9 X8 X7 X6 X5 X4 X3 X2 X1 X0 8202H X1F X1E X1D X1C X1B X1A X19 X18 X17 X16 X15 X14 X13 X12 X11 X10 8204H X2F X2E X2D X2C X2B X2A X29 X28 X27 X26 X25 X24 X23 X22 X21 X20 • Stores ON/OFF data from an output unit, read only. • 0 indicates ON and 1 OFF. The output memory is accessed as shown below? Write, Read, Output unit, Output refresh after END instruction is executed, Output memory Direct mode Refresh mode
Internal relay (M) Latch relay (L) Step relay (S)		8400H to 84FFH: M/L/S 0 to 2047	○ Stores device ON/OFF data in one bit locations. ○ 0 indicates OFF and 1 ON. Example: M0 to 47 are as follows: Odd address / Even address B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 8400H M15 M14 M13 M12 M11 M10 M9 M8 M7 M6 M5 M4 M3 M2 M1 M0 8402H M31 M30 M29 M28 M27 M26 M25 M24 M23 M22 M21 M20 M19 M18 M17 M16 8404H M47 M46 M45 M44 M43 M42 M41 M40 M39 M38 M37 M36 M35 M34 M33 M32 • Stores PC operation results and allows read/write.
Link relay (B)		8600H to 867FH: B0 to 3FF	
Annunciator (F)		8700H to 871FH: F0 to 255	

IB (NA) 66147-A

Device	CPU Type	Address	Configuration
Special relay (M)	A3H	8740H to 875FH — M9000 to 9255	○ Stores device ON/OFF data in one bit locations. ○ 0 indicates OFF and 1 ON. Example: M0 to 47 are as follows:
Timer (T) contact		8780H to 879FH — T0 to 255	
Counter (C) contact		87C0H to 87DFH — C0 to 255	
Timer (T) coil		9C00H to 9C1FH — T0 to 255	• Stores PC operation results and allows read/write.
Counter (C) coil		9C40H to 9C5FH — C0 to 255	

	Odd address								Even address							
	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
8400H	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0
8402H	M31	M30	M29	M28	M27	M26	M25	M24	M23	M22	M21	M20	M19	M18	M17	M16
8404H	M47	M46	M45	M44	M43	M42	M41	M40	M39	M38	M37	M36	M35	M34	M33	M32

Device	CPU Type	Address	Configuration
Data register (D)	A3H	8800H to 8FFFH: D0 to 1023	All devices consist of two bytes (16 bits). Example The configuration of D0 is as shown below: B7 B0 8800H (L) 8801H (H) B15 B8
Link register (W)		9000H to 97FFH: W0 to 3FF	
Timer (T) present value		9800H to 99FFH: T0 to 255	
Counter (C) present value		9A00H to 9BFFH: C0 to 255	
Special register (D)		9D00H to 9EFFH: D9000 to 9255	
Accumulator (A0, 1)		9FF8H: A0 9FFAH: A1	
Index (Z, V)		9FFCH: Z 9FFFH: V	

APPENDIX 1 Operation Processing Time

The operation processing time of each instruction is shown in the tables on the following pages.
The operation processing time differs depending on values in the source and destination. Use the values in the tables as a guide to processing time.

(1) Processing time varies depending on the I/O control mode used with any instruction operating on inputs or outputs.

(2) The processing time for each instruction is shown for refresh mode.
The refresh processing time after END can be calculated as follows:

Sequence program processing time =
(instruction processing time) + (END processing time) + (refresh processing time)

Obtained from the list

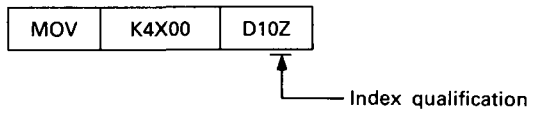
END processing time =
(END instruction processing time) + (T/C processing time at END)

Refresh processing time = (input + output point)/16 $\times \frac{5.4\ (A1N\ to\ A3N)}{4.375\ (A3H)}$ (μs)

(3) The following processings may take a slightly longer period of time.

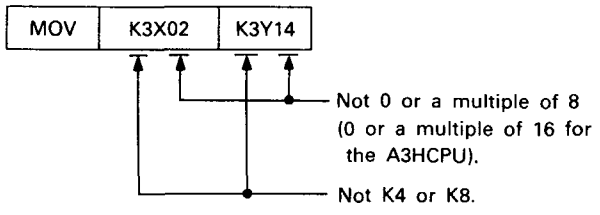
1) Device specified indirectly as source or destination is used with the index register (V, Z).

Example:



2) The number of digits specified for the devices used with any basic or application instruction is not K4 or K8 and/or the device number specified is not 0 or a multiple of 8 (0 or a multiple of 16 when the A3HCPU is used).

Example:



APP

1.1 Sequence instructions

Instruction	Condition (Device)				Processing Time (μ sec)			
					A1N, A2N, A3N		A3H	
					Refresh mode	Direct mode	Refresh mode	Direct mode
LD LDI AND	X				1.0	2.3	0.2	2.0
ANI OR ORI	Y, M. L, S, B, F, T, C				1.0		0.2	
ANB	———							
ORB	———							
OUT	Y	Unchanged (OFF→OFF, ON→ON)			1.0	2.3	0.35	
		Changed (OFF→ON, ON→OFF)			1.0	2.3	0.4	2.0
	M (other than special M), L, S, B	Unchanged (OFF→OFF, ON→ON)			1.0		0.35	
		Changed (OFF→ON, ON→OFF)			1.0		0.4	
	Special M				37.1		1.4	
	F	Unexecuted			61.1		62	
		Executed			663.1		283	
	T	Instruction execution time			1.0		0.2	
		END processing time	Unexecuted		0		0	
			Executed	After time out	11.2		3.7	
				Added K	23.8		5.9	
				Added D	29.5		6.5	
	C	Instruction execution time			1.0		0.2	
		END processing time	Unexecuted		0		0	
			Executed	Uncounted	0		0	
				After count out	0		0	
				Counted K	25.1		3.8	
				Counted D	29.5		4.6	
SET	Y	Unexecuted			1.0	2.3	0.35	
		Executed	Unchanged (ON→ON)		1.0	2.3	0.35	
			Changed (OFF→ON)		1.0	2.3	0.4	2.0
	M, L, S, B	Unexecuted			1.0		0.35	
		Executed	Unchanged (ON→ON)		1.0		0.35	
			Changed (OFF→ON)		1.0		0.4	
	Special M B	Unexecuted			3.0		0.8	
		Executed			32.3		1.4	
	F	Unexecuted			3.0		0.8	
		Executed			637.7		283	

APP

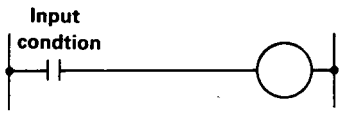
Instruction	Condition (Device)			Processing Time (μsec)			
				A1N, A2N, A3N		A3H	
				Refresh mode	Direct mode	Refresh mode	Direct mode
RST	Y		Unexecuted	1.0	2.3	0.35	
		Executed	Unchanged (ON→ON)	1.0	2.3	0.35	
			Changed (OFF→ON)	10.	2.3	0.4	2.0
	M, L, S, B		Unexecuted	1.0		0.35	
		Executed	Unchanged (ON→ON)	1.0		0.35	
			Changed (OFF→ON)	1.0		0.4	
	Special M, B		Unexecuted	3.0		0.8	
			Executed	32.1		1.4	
	F		Unexecuted	3.0		0.8	
			Executed	477.1		427	
	T, C		Unexecuted	3.0		0.8	
			Executed	43.1		5.2	
	D, W, A0, A1, V, Z		Unexecuted	3.0		0.8	
			Executed	27.5		0.8	
	R		Unexecuted	3.0		0.8	
			Executed	34.6		56.5	
NOP	———			1.0		0.2	
FEND	M9084 off			2150		1128	
END	M9084 on			2060		988	
MC	Y		Unexecuted	43.1	44.4	2.6	6.4
			Executed	39.4	40.7	2.6	6.4
	M, L, S, B, F		Unexecuted	43.1		2.6	
			Executed	39.4		2.6	
MCR	———			26.4		1.2	
PLS	Y		Unexecuted	59.3	60.6	1.8	5.6
		Executed	On	61.9	63.2	1.8	5.6
			Off	60.3	61.6	1.8	5.6
PLF	M, L, B, F		Unexecuted	59.1		1.8	
		Executed	On	62.2		1.8	
			Off	60.6		1.8	
SFT	Y		Unexecuted	3.0	3.0	0.8	0.8
			Executed	37.6	39.4	9.1	10.9
SFTP	M, L, B, F		Unexecuted	3.0		0.8	
			Executed	37.6		9.1	
MPS	———			1.0		0.2	
MRD	———						
MPP	———						
CJ	Without index qualification			39.2		4.0	
	With index qualification			47.7		7.2	

APP

Instruction	Condition (Device)	Processing Time (μsec)			
		A1N, A2N, A3N		A3H	
		Refresh mode	Direct mode	Refresh mode	Direct mode
SCJ	Without index qualification	71.2		4.0	
	With index qualification	80.5		7.2	
JMP		39.2		3.8	
CALL	Without index qualification	74		8.2	
	With index qualification	78.2		11.8	
CALLP	Without index qualification	69.7		8.2	
	With index qualification	78.2		11.8	
RET		50.2		5.8	
EI		37.7		53	
DI		65.5		52.5	
IRET		119.7		61.6	
SUB	Without index qualification	78.7		86	
	With index qualification	85.2		88	
SUBP	Without index qualification	78.7		86	
	With index qualification	85.2		88	
CHG	M9084 off	2420		1128	
	M9084 on	2340		988	
FOR		53.2		5.8	
NEXT		40.7		6.4	
STOP		———		———	

POINT

(1) "When not executed" in the above table indicates that the input condition is off.



(2) "When not counted" of OUT C instruction indicates that the input condition remains on and the counter does not count.

(3) "OFF" of PLS and PLF instructions indicates that the input condition remains on 1 scan after it has turned on (off for PLF), and the pulse is not generated.

(4) T/C count processing time and refresh time are not included in the FEND, END, CHG instruction processing times.

APP

1.2 Basic Instructions

Instruction	Condition	Processing Time (μsec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
LD= (S1) (S2)		70	70.4	87.4	2.8	2.8	10.4
AND= (S1) (S2)		61	61.9	80.9	1.8	1.8	9.4
OR= (S1) (S2)		67	66.4	84.9	3.2	3.2	10.8
LDD= (S1) (S2)		133	134.4	119.4	157	157	180
ANDD= (S1) (S2)		124	124.9	210.4	157	157	180
ORD= (S1) (S2)		133	133.4	218.4	158	158	181
LD<> (S1) (S2)		69	69.4	86.4	2.8	2.8	10.4
AND<> (S1) (S2)		60	59.9	79.4	1.8	1.8	9.4
OR<> (S1) (S2)		66	65.9	83.9	3.2	3.2	10.8
LDD<> (S1) (S2)		131	132.4	217.4	158	158	181
ANDD<> (S1) (S2)		129	129.4	215.4	158	158	181
ORD<> (S1) (S2)		129	128.4	214.4	161	161	184
LD> (S1) (S2)		67	67.4	84.4	2.8	2.8	10.4
AND> (S1) (S2)		60	59.9	78.9	1.8	1.8	9.4
OR> (S1) (S2)		66	64.9	83.4	3.2	3.2	10.8
LDD> (S1) (S2)		133	133.4	219.4	158	158	181
ANDD> (S1) (S2)		131	131.4	217.4	158	158	181
ORD> (S1) (S2)		131	130.4	219.4	161	161	184
LD<= (S1) (S2)		71	70.9	87.9	2.8	2.8	10.4
AND<= (S1) (S2)		61	61.4	80.9	1.8	1.8	9.4
OR<= (S1) (S2)		69	67.9	86.4	3.2	3.2	10.8
LDD<= (S1) (S2)		137	136.4	222.4	158	158	181
ANDD<= (S1) (S2)		128	128.4	213.4	158	158	182
ORD<= (S1) (S2)		137	136.4	221.4	161	161	184

APP

Instruction	Condition	Processing Time (μ sec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
LD< (S1) (S2)		69	69.4	86.4	2.8	2.8	10.4
AND< (S1) (S2)		59	59.9	78.9	1.8	1.8	9.4
OR< (S1) (S2)		66	65.4	83.9	3.2	3.2	10.8
LDD< (S1) (S2)		133	133.4	219.4	159	159	182
ANDD< (S1) (S2)		131	131.4	217.4	158	158	181
ORD< (S1) (S2)		131	130.4	215.4	160	160	183
LD>= (S1) (S2)		70	70.9	87.9	2.8	2.8	10.4
AND>= (S1) (S2)		61	61.4	80.4	1.8	1.8	9.4
OR>= (S1) (S2)		69	68.4	86.4	3.2	3.2	10.8
LDD>= (S1) (S2)		137	137.4	222.4	160	160	181
ANDD>= (S1) (S2)		127	128.4	213.4	158	158	181
ORD>= (S1) (S2)		137	136.4	221.4	161	161	183
+ (S) (D)		44	44.8	58.9	1.6	1.6	9.2
+P (S) (D)		44	44.8	58.9	1.6	1.6	9.2
D+ (S) (D)		69	69.4	90.4	3.0	3.0	18.2
D+P (S) (D)		69	69.4	90.4	3.0	3.0	18.2
+ (S1) (S2) (D)		77	77.4	103.4	1.8	1.8	13.2
+P (S1) (S2) (D)		77	77.4	103.4	1.8	1.8	13.2
D+ (S1) (S2) (D)		99	99.4	246.4	3.0	3.0	25.8
D+P (S1) (S2) (D)		99	99.4	246.4	3.0	3.0	25.8
− (S) (D)		45	45.4	59.4	1.6	1.6	9.2
−P (S) (D)		45	45.4	59.4	1.6	1.6	9.2
D− (S) (D)		69	69.4	90.4	3.0	3.0	18.2
D−P (S) (D)		69	69.4	90.4	3.0	3.0	18.2

APP

Instruction	Condition	Processing Time (μ sec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
− (S1) (S2) (D)		79	79.4	106.9	1.8	1.8	13.2
−P (S1) (S2) (D)		79	79.4	106.9	1.8	1.8	13.2
D− (S1) (S2) (D)		99	99.4	130.4	3.0	3.0	25.8
D−P (S1) (S2) (D)		99	99.4	130.4	3.0	3.0	25.8
✳ (S1) (S2) (D)		94	94.9	168.4	2.4	2.4	17.6
✳P (S1) (S2) (D)		94	94.9	168.4	2.4	2.4	17.6
D✳ (S1) (S2) (D)		341	340.4	370.4	18.2	18.2	41
D✳ (S1) (S2) (D)		341	340.4	370.4	18.2	18.2	41
/ (S1) (S2) (D)		102	102.9	98.9	8.6	8.6	20
/P (S1) (S2) (D)		102	102.9	98.9	8.6	8.6	20
D/ (S1) (S2) (D)		393	394.4	412.4	36.8	36.8	59.6
D/P (S1) (S2) (D)		393	394.4	412.4	36.8	36.8	59.6
INC (D)		29	29.4	37.6	1.2	1.2	5
INCP (D)		29	29.4	37.6	1.2	1.2	5
DINC (D)		42	42	132.4	2.2	2.2	9.8
DINCP (D)		42	42	132.4	2.2	2.2	9.8
DEC (D)		31	31.2	39.4	1.2	1.2	5
DECP (D)		31	31.2	39.4	1.2	1.2	5
DDEC (D)		42	42	53.9	2.2	2.2	9.8
DDECP (D)		42	42	53.9	2.2	2.2	9.8
B+ (S) (D)		123	123.4	183.4	3.6	3.6	11.2
B+P (S) (D)		123	123.4	183.4	3.6	3.6	11.2
DB+ (S) (D)		175	176.4	280.4	46.8	46.8	62
DB+P (S) (D)		175	176.4	280.4	46.8	46.8	62

APP

Instruction	Condition	Processing Time (μsec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
B+ (S1) (S2) (D)		129	129.4	192.4	23	23	34.4
B+P (S1) (S2) (D)		129	129.4	192.4	23	23	34.4
DB+ (S1) (S2) (D)		187	186.4	294.4	274	274	308
DB+P (S1) (S2) (D)		187	186.4	294.4	274	274	308
B− (S) (D)		125	124.9	185.4	3.6	3.6	11.2
B−P (S) (D)		125	124.9	185.4	3.6	3.6	11.2
DB− (S) (D)		175	175.4	280.4	46.8	46.8	6.2
DB−P (S) (D)		175	175.4	280.4	46.8	46.8	6.2
B− (S1) (S2) (D)		133	133.4	203.4	23	23	34.4
B−P (S1) (S2) (D)		133	133.4	203.4	23	23	34.4
DB− (S1) (S2) (D)		185	186.4	294.4	261	261	306
DB−P (S1) (S2) (D)		185	186.4	294.4	261	261	306
B* (S1) (S2) (D)		299	300.4	358.4	10.8	10.8	22.2
B*P (S1) (S2) (D)		299	300.4	358.4	10.8	10.8	22.2
DB* (S1) (S2) (D)		941	939.4	1044.4	693	693	738
DB*P (S1) (S2) (D)		941	939.4	1044.4	693	693	738
B/ (S1) (S2) (D)		235	236.4	274.4	25.2	25.2	40.4
B/P (S1) (S2) (D)		235	236.4	274.4	25.2	25.2	40.4
DB/ (S1) (S2) (D)		896	894.4	954.4	748	748	793
DB/P (S1) (S2) (D)		896	894.4	954.4	748	748	793
BCD (S) (D)		82	82.9	90.4	1.6	1.6	9.2
BCDP (S) (D)		82	82.9	90.4	1.6	1.6	9.2
DBCD (S) (D)		219	220.4	284.4	9.4	9.4	24.6
DBCDP (S) (D)		219	220.4	284.4	9.4	9.4	24.6

APP

Instruction	Condition	Processing Time (μsec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
BIN Ⓢ Ⓓ		79	78.4	86.4	1.6	1.6	9.2
BINP Ⓢ Ⓓ		79	78.4	86.4	1.6	1.6	9.2
DBIN Ⓢ Ⓓ		215	216.4	280.4	3.6	3.6	18.8
DBINP Ⓢ Ⓓ		215	216.4	280.4	3.6	3.6	18.8
MOV Ⓢ Ⓓ		47	47.2	56.9	1.2	1.2	8.8
MOVP Ⓢ Ⓓ		47	47.2	56.9	1.2	1.2	8.8
DMOV Ⓢ Ⓓ		67	66.9	86.9	2.0	2.0	17.2
DMOVP Ⓢ Ⓓ		67	66.9	86.9	2.0	2.0	17.2
XCH (D1) (D2)		60	60.9	83.9	1.8	1.8	9.4
XCHP (D1) (D2)		60	60.9	83.9	1.8	1.8	9.4
DXCH (D1) (D2)		107	107.4	141.4	3.6	3.6	18.8
DXCHP (D1) (D2)		107	107.4	141.4	3.6	3.6	18.8
CML Ⓢ Ⓓ		43	43.4	57.4	1.4	1.4	9.0
CMLP Ⓢ Ⓓ		43	43.4	57.4	1.4	1.4	9.0
DCML Ⓢ Ⓓ		74	74.9	108.4	2.6	2.6	17.8
DCMLP Ⓢ Ⓓ		74	74.9	108.4	2.6	2.6	17.8
BMOV Ⓢ Ⓓ n	n=96	399	400.4	7144.4	132	132	862
BMOVP Ⓢ Ⓓ n	n=96	399	400.4	7144.4	132	132	862
FMOV Ⓢ Ⓓ n	n=96	229	228.4	1029.4	66	66	435
FMOVP Ⓢ Ⓓ n	n=96	229	228.4	1029.4	66	66	435

POINT

(1) All the basic instructions indicated above are used without index qualification.

(2) When unexecuted, any instruction is processed during the following time:

A1N, A2N, A3NCPU ..... (Number of steps + 1) × 1.0 (μs)
A3HCPU ..... (Number of steps + 1) × 0.2 (μs)

APP

1.3 Application Instructions

Instruction	Condition	Processing Time (μ sec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
WAND Ⓢ Ⓓ		59.5	58.7	71.7	1.6	1.6	9.2
WANDP Ⓢ Ⓓ		59.5	58.7	71.2	1.6	1.6	9.2
DAND Ⓢ Ⓓ		139.5	138.7	239.7	27.4	27.4	42.6
DANDP Ⓢ Ⓓ		139.5	138.7	239.7	27.4	27.4	42.6
WAND S1 S2 Ⓓ		95.5	95.7	151.7	20.8	20.8	32.2
WANDP S1 S2 Ⓓ		95.5	95.7	151.7	20.8	20.8	32.2
WOR Ⓢ Ⓓ		60.5	60.2	72.2	1.6	1.6	9.2
WORP Ⓢ Ⓓ		60.5	60.2	72.2	1.6	1.6	9.2
DOR Ⓢ Ⓓ		139.5	138.7	239.7	27.4	27.4	42.6
DORP Ⓢ Ⓓ		139.5	138.7	239.7	27.4	27.4	42.6
WOR S1 S2 Ⓓ		96.5	95.7	151.7	20.8	20.8	32.2
WORP S1 S2 Ⓓ		96.5	95.7	151.7	20.8	20.8	32.2
WXOR Ⓢ Ⓓ		59.5	58.7	71.7	1.6	1.6	9.2
WXORP Ⓢ Ⓓ		59.5	58.7	71.7	1.6	1.6	9.2
DXOR Ⓢ Ⓓ		139.5	138.7	239.7	27.4	27.4	42.6
DXORP Ⓢ Ⓓ		139.5	138.7	239.7	27.4	27.4	42.6
WXOR S1 S2 Ⓓ		96.5	95.7	151.7	20.8	20.8	32.2
WXORP S1 S2 Ⓓ		96.5	95.7	151.7	20.8	20.8	32.2
WXNR Ⓢ Ⓓ		63.5	62.2	73.7	1.6	1.6	9.2
WXNRP Ⓢ Ⓓ		63.5	62.2	73.7	1.6	1.6	9.2
DXNR Ⓢ Ⓓ		141.5	139.7	240.7	27.4	27.4	42.6
DXNRP Ⓢ Ⓓ		141.5	139.7	240.7	27.4	27.4	42.6
WXNR S1 S2 Ⓓ		97.5	96.2	151.7	20.8	20.8	32.2
WXNRP S1 S2 Ⓓ		97.5	96.2	151.7	20.8	20.8	32.2

APP

Instruction		Condition	Processing Time (μ sec)					
			A1N, A2N, A3N			A3H		
			Refresh mode	Direct mode		Refresh mode	Direct mode	
			All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
NEG	Ⓓ		49.5	48.7	85.7	14.4	14.4	18.2
NEGP	Ⓓ		49.5	48.7	85.7	14.4	14.4	18.2
ROR	n	A1N to A3N :n=3 A3H :n=5	51.5	51.2		4.8	4.8	
RORP	n	A1N to A3N :n=3 A3H :n=5	51.5	51.2		4.8	4.8	
RCR	n	A1N to A3N :n=3 A3H :n=5	58.5	58.7		6.8	6.8	
RCRP	n	A1N to A3N :n=3 A3H :n=5	58.5	58.7		6.8	6.8	
ROL	n	A1N to A3N :n=3 A3H :n=5	53.5	53.2		4.6	4.6	
ROLP	n	A1N to A3N :n=3 A3H :n=5	53.5	53.2		4.6	4.6	
RCL	n	A1N to A3N :n=3 A3H :n=5	56.5	56.7		6.8	6.8	
RCLP	n	A1N to A3N :n=3 A3H :n=5	56.5	56.7		6.8	6.8	
DROR	n	A1N to A3N :n=3 A3H :n=5	69.5	69.2		10.6	10.6	
DRORP	n	A1N to A3N :n=3 A3H :n=5	69.5	69.2		10.6	10.6	
DRCR	n	A1N to A3N :n=3 A3H :n=5	71.5	72.2		13.0	13.0	
DRCRP	n	A1N to A3N :n=3 A3H :n=5	71.5	72.2		13.0	13.0	
DROL	n	A1N to A3N :n=3 A3H :n=5	69.5	69.2		10.6	10.6	
DROLP	n	A1N to A3N :n=3 A3H :n=5	69.5	69.2		10.6	10.6	

Instruction	Condition	Processing Time (μ sec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
DRCL n	A1N to A3N :n=3 A3H :n=5	67.5	67.7		13.0	13.0	
DRCLP n	A1N to A3N :n=3 A3H :n=5	67.5	67.7		13.0	13.0	
SFR Ⓓ n	A1N to A3N :n=5 A3H :n=5	73.5	72.2	82.7	4.0	4.0	7.8
SFRP Ⓓ n	A1N to A3N :n=5 A3H :n=5	73.5	72.2	82.7	4.0	4.0	7.8
BSFR Ⓓ n	A1N to A3N :n=5 A3H :n=5	123.5	122.7	123.7	116	116	154
BSFRP Ⓓ n	A1N to A3N :n=5 A3H :n=5	123.5	122.7	123.7	116	116	154
DSFR Ⓓ n	A1N to A3N :n=5 A3H :n=5	117.5	115.7	—	15.2	15.2	—
DSFRP Ⓓ n	A1N to A3N :n=5 A3H :n=5	117.5	115.7	—	15.2	15.2	—
SFL Ⓓ n	A1N to A3N :n=5 A3H :n=5	73.5	72.7	83.7	4.0	4.0	7.8
SFLP Ⓓ n	A1N to A3N :n=5 A3H :n=5	73.5	72.7	83.7	4.0	4.0	7.8
BSFL Ⓓ n	A1N to A3N :n=5 A3H :n=5	133.5	132.7	133.7	116	116	154
BSFLP Ⓓ n	A1N to A3N :n=5 A3H :n=5	133.5	132.7	133.7	116	116	154
DSFL Ⓓ n	A1N to A3N :n=5 A3H :n=5	117.5	116.7	—	15.8	15.8	—
DSFLP Ⓓ n	A1N to A3N :n=5 A3H :n=5	117.5	116.7	—	15.8	15.8	—
SER (S1) (S2) n	A1N to A3N :n=5 A3H :n=5	199.5	199.7	—	187	187	—
SERP (S1) (S2) n	A1N to A3N :n=5 A3H :n=5	199.5	199.7	—	187	187	—

APP

Instruction	Condition	Processing Time (μ sec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
SUM Ⓢ		114.5	114.2	130.7	14.4	14.4	18.2
SUMP Ⓢ		114.5	114.2	130.7	14.4	14.4	18.2
DSUM Ⓢ		199.5	198.7	230.7	33.8	33.8	37.6
DSUMP Ⓢ		199.5	198.7	230.7	33.8	33.8	37.6
DECO Ⓢ Ⓓ n	A1N to A3N :n=2 A3H :n=2	163.5	162.7	215.7	200	200	205
DECOP Ⓢ Ⓓ n	A1N to A3N :n=2 A3H :n=2	163.5	162.7	215.7	200	200	205
SEG Ⓢ Ⓓ		90.7	90.7	154.7	3.4	3.4	11.0
ENCO Ⓢ Ⓓ n	A1N to A3N :n=2 A3H :n=2	163.5	162.7	194.7	188	188	193
ENCOP Ⓢ Ⓓ n	A1N to A3N :n=2 A3H :n=2	163.5	162.7	194.7	188	188	193
BSET Ⓓ n	A1N to A3N :n=5 A3H :n=5	89.5	89.7	—	5.0	5.0	—
BSETP Ⓓ n	A1N to A3N :n=5 A3H :n=5	89.5	89.7	—	5.0	5.0	—
BRST Ⓓ n	A1N to A3N :n=5 A3H :n=5	96.5	95.7	—	5.0	5.0	—
BRSTP Ⓓ n	A1N to A3N :n=5 A3H :n=5	96.5	95.7	—	5.0	5.0	—
UNI Ⓢ Ⓓ n	A1N to A3N :n=4 A3H :n=4	130.5	130.7	—	155	155	—
UNIP Ⓢ Ⓓ n	A1N to A3N :n=4 A3H :n=4	130.5	130.7	—	155	155	—
DIS Ⓢ Ⓓ n	A1N to A3N :n=4 A3H :n=4	153.5	152.7	—	155	155	—
DISP Ⓢ Ⓓ n	A1N to A3N :n=4 A3H :n=4	153.5	152.7	—	155	155	—

APP

Instruction	Condition	Processing Time (μ sec)						
		A1N, A2N , A3N				A3H		
		Refresh mode		Direct mode		Refresh mode	Direct mode	
		Other than X, Y	X, Y	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
ASC ASCII Character Ⓓ		120				107		
FIFW Ⓢ Ⓓ		100.5		100.7	123.2	136	136	140
FIFWP Ⓢ Ⓓ		100.5		100.7	123.2	136	136	140
FIFR (D1) (D2)		117.5		117.7	133.7	207	207	211
FIFRP (D1) (D2)		117.5		117.7	133.7	207	207	211
FROM n1 n2 Ⓓ n3 When AD71	n3=1	439	524	427.7	507.7	300		
	n3=1000	6609	2109 n3=112	6597.7	2357.7 n3=112	5050		
FROMP n1 n2 Ⓓ n3 When AD71	n3=1	439	524	427.7	507.7	300		
	n3=1000	6609	2109 n3=112	6597.7	2357.7 n3=112	5050		
DFRO n1 n2 Ⓓ n3 When AD71	n3=1	449	529	432.7	517.7	300		
	n3=500	6609	2109 n3=56	6597.7	1867.7 n3=56	5050		
DFROP n1 n2 Ⓓ n3 When AD71	n3=1	449	529	432.7	517.7	300		
	n3=500	6609	2109 n3=56	6597.7	1867.7 n3=56	5050		
TO n1 n2 Ⓢ n3 When AD71	n3=1	449	539	432.7	522.7	300		
	n3=1000	6609	3909 n3=112	6597.7	3917.7 n3=112	5050		
TOP n1 n2 Ⓢ n3 When AD71	n3=1	449	539	432.7	522.7	300		
	n3=1000	6609	3909 n3=112	6597.7	3917.7 n3=112	5050		
DTO n1 n2 Ⓓ n3 When AD71	n3=1	454	544	442.7	517.7	300		
	n3=500	6609	1609 n3=56	6597.7	1597.7 n3=56	5050		
DTOP n1 n2 Ⓓ n3 When AD71	n3=1	454	544	442.7	517.7	300		
	n3=500	6609	1609 n3=56	6597.7	1597.7 n3=56	5050		
LRDP n1 Ⓢ Ⓓ n2	n3=1	189.7				228		
	n3=32	189.7				228		
LWTP n1 Ⓓ Ⓢ n2	n3=1	199.7				236		
	n3=32	445.7				415		

APP

Instruction	Condition	Processing Time (μ sec)					
		A1N, A2N, A3N			A3H		
		Refresh mode	Direct mode		Refresh mode	Direct mode	
		All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
RFRP n1 n2 Ⓓ n3	n3=1	171.7			183		
	A1N to A3N :n3=16 A3H :n3=32	171.7			183		
RTOP n1 n2 Ⓓ n3	n3=1	175.7			185		
	A1N to A3N :n3=16 A3H :n3=32	175.7			185		
WDT		63.7			49.3		
WDTP		63.7			49.3		
CHK (D1) (D2) Fault check instruction	1 condition contact	—	771		281.5		
	50 condition contact	—	3380		2210		
	100 condition contact	—	6887		4180		
	150 condition contact	—	10137		6140		
SLT	Device memory only	8447.7			4100		
	Device memory + R (8K points)	24597.7			10400		
SLTR		29.3			52.6		
STRA		30.2			51.8		
STRAR		27.7			51.8		
STC		28.1			1.2		
CTC		30.5			1.2		
DUTY n1 n2 Ⓓ		67.7			121		
PR Ⓢ Ⓓ		225.7			183		
PRC Ⓢ Ⓓ		140.7			145		
CHK (D1) (D2) Bit reverse output instruction		120.5	—		—		

APP

Instruction		Condition	Processing Time (μsec)					
			A1N, A2N, A3N			A3H		
			Refresh mode	Direct mode		Refresh mode	Direct mode	
			All devices	Other than X, Y	X, Y	All devices	Other than X, Y	X, Y
LED	Ⓢ		202.7			282		
LEDC	Ⓢ		264.7			320		
LEDA	ASCII Character		201.7			262		
LEDB	ASCII Character		210.7			262		
LEDR			637.7			460		

POINT

(1) All the basic instructions indicated above are used without index qualification.

(2) When unexecuted, any instruction is processed during the following time:

A1N, A2N(S1), A3NCPU ..... (Number of steps + 1) × 1.0 (μs)
A3HCPU ..... (Number of steps + 1) × 0.2 (μs)

APP

IB (NA) 66147-A

APPENDIX 2 Self-Diagnosis

When error has occurred during the run of programmable controller, the self-diagnostic function stops the CPU and provides error display, etc. The contents of self-diagnostic function are indicated in Table 2.8.
For the operations inside the CPU and error resetting procedures according to errors, refer to Appendix 3 "ERROR CODE LIST".

Diagnosis		CPU Status	"RUN" LED Status	Error Display		Applicable CPU type			
				A1(E), A2(E)CPU "ERROR" display	A3(E)CPU LED display message	A1N	A2N	A3N	A3H
Memory error	Instruction code check	Stop	☆ Flicker	On	INSTRCT. CODE ERR.	●	●	●	●
	Parameter setting check			On	PARAMETER ERROR	●	●	●	●
	No END instruction			On	MISSING END INS.	●	●	●	●
	Instruction execution disable			On	CAN'T EXECUTE	●	●	●	●
	Instruction execution disable			On	CAN'T EXECUTE (I)	●	●	●	●
	No memory cassette			On	CASSETTE ERROR	—	●	●	●
CPU error	RAM check	Stop	☆ Flicker	On	RAM ERROR	●	●	●	●
	Operation circuit check			On	OPE. CIRCUIT ERR.	●	●	●	●
	Watch dog error monitor			On	WDT ERROR	●	●	●	●
	Sub CPU check			—	SUB-CPU ERROR	—	—	—	●
	END instruction unexecution			On	END NOT EXECUTE	●	●	●	●
	Endless loop			—	WDT ERROR	●	●	●	●
	Main CPU error			—	WDT ERROR	—	—	—	●
	Interrupt error	Stop	Off	—	MAIN CPU DOWN	—	—	—	●
I/O error	I/O module verify	Stop / Run	Flicker / On	On	UNIT VERIFY ERR.	●	●	●	●
	Fuse blown			On	FUSE BREAK OFF	●	●	●	●
Special function unit error	Control bus check	Stop	☆ Flicker	On	CONTROL-BUS ERR.	●	●	●	●
	Special function module error			On	SP. UNIT DOWN	●	●	●	●
	Link module error			On	LINK UNIT ERROR	●	●	●	●
	I/O interruption error			On	I/O INT. ERROR	●	●	●	●
	Special function module assignment error			On	SP. UNIT LAY. ERR.	●	●	●	●
	Special function module error	Stop / Run	Flicker ☆ / On	On	SP. UNIT ERROR	●	●	●	●
	Link parameter error	Run	On	On	LINK PARA. ERROR	●	●	●	●
Battery	Battery low	Run	On	On	BATTERY ERROR	●	●	●	●
*Operation check error		Stop / Run	Flicker ☆ / On	On	OPERATION ERROR	●	●	●	●
Main CPU check		Stop	Flicker	—	MAIN-CPU DOWN	—	—	—	—

Table 2.1 Self-Diagnosis List

APP

REMARKS

1. Two modes described in the "CPU Status" and "RUN LED Status" columns in Table 2.1 indicate that they can be changed by the setting of peripheral unit.

2. When the CPU is A3N, A3HCPU, the status of "RUN" LED with ☆ mark is "off".

APPENDIX 3 Error Code List

When error occurs in the RUN mode or during the operation of PC, an error display or error code (including a step number) is stored to the special register by the self-diagnostic function. The error code reading procedure and the causes and corrective actions of errors at the occurrence of error are shown in Table 3.1. Take proper corrective action and remove the cause of error.

3.1 Error code reading procedure

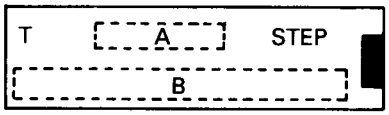
When error has occurred, the error code can be read by the PU or GPP/GHP/PHP. The procedures are described below.

(1) PU

Operation [TEST] [DEC K] [STP +] or [TEST] [DEC K] [STP −]

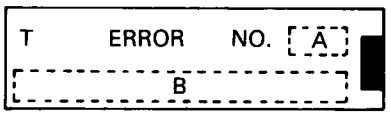
By this key operation, the following display is provided on the liquid crystal display screen.

■ When the error code is 10, 13, 46 or 50



A: Error step number (Maximum of 5 digits)
B: Error message

■ When the error number is other than the above



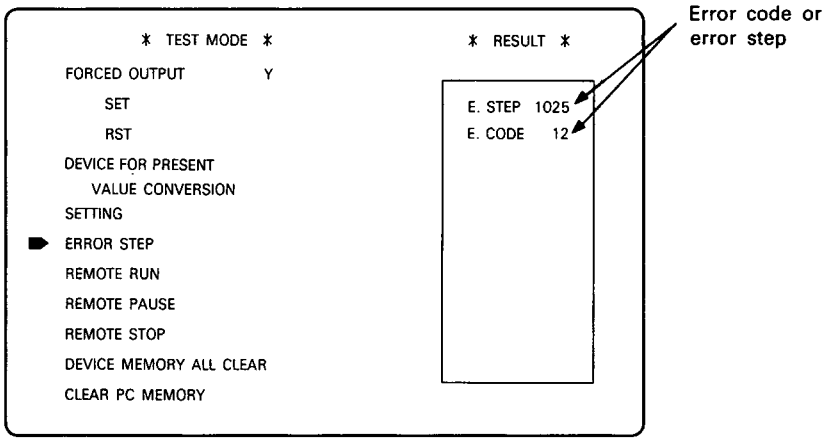
A: Error code
B: Error message

APP

(2) GPP/GHP/PHP

Operation [PC] [TEST] [↓] [GO] ------- [GO]

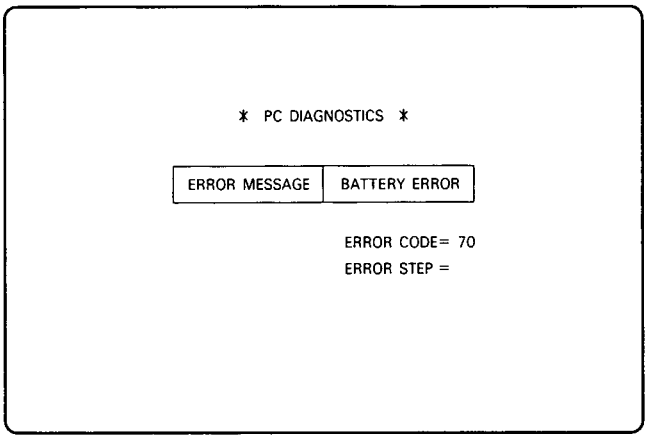
By pressing the [GO] key, the error codes are displayed in sequence. Press the [↓] key until the cursor is brought to the position of "error step" item.



When the error code is 10, 13, 46, or 50, the error step number is displayed. For the error code, perform PC diagnosis in the procedure indicated below to make user of the error message corresponding to the error code.

Operation [LST] [MNT] [↓] [GO]

Move the cursor to the position of "PC DIAGNOSIS".



Another method to confirm the error code is to monitor the contents of special register D9008 by use of the PU or GPP/GHP/PHP.

3.2 Error code list

This section describes the contents, causes, and corrective actions of error numbers and error messages.

Error Message	Content of Special Register D9008 (BIN value)	CPU States	Error and Cause	Corrective Action
"INSTRCT. CODE ERR" (Checked at the execution of instruction)	10	Stop	Instruction code, which cannot be decoded by CPU, is included in the program. (1) ROM including instruction code, which cannot be decoded, has been loaded. (2) Since the memory contents have changed for some reason, instruction code, which cannot be decoded, has been included.	(1) Read the error step by use of a peripheral equipment and correct the program at that step. (2) In the case of ROM, rewrite the contents of change the ROM with a ROM which stores correct contents.
"PARAMETER ERROR" (Checked at power-on, STOP → RUN, and PAUSE → RUN)	11	Stop	(1) Capacity larger than the memory capacity of CUP has been set and then write to CPU has been performed. (2) The contents of parameters of CPU memory have changed due to noise or the improper loading of memory.	(1) Check the memory capacity of CPU with the memory capacity set by peripheral equipment and re-set incorrect area by the peripheral equipment. (2) Check the loading of CPU memory and load it correctly. (3) Read the parameter contents of CPU memory, check and correct the contents, and write them to the memory again.
"MISSING END INS." (Checked at STOP → RUN)	12	Stop	(1) There is no END (FEND) instruction in the program. (2) When subprogram has been set by the parameter, there is no END instruction in the subprogram.	Write END at the end of program.
"CAN'T EXECUTE (P)" (Checked at the execution of instruction)	13	Stop	(1) There is no jump destination or multiple destinations specified by the [CJ], [SCJ], [CALL], [CALLP], or [JMP] instruction. (2) There is a [CHG] instruction and no setting of subprogram. (3) Although there is no [CALL] instruction, the [RET] instruction exists in the program and has been executed. (4) The [CJ], [SCJ], [CALL], [CALLP], or [JMP] instruction has been executed with its jump destination located below the END instruction.	Read the error step by use of peripheral equipment and correct the program at that step. (Make correction such as the insertion of jump destination or the changing of jump destinations to one.)

Table 3.1 Error Code List (Continue)

APP

Error Message	Content of Special Register D9008 (BIN value)	CPU States	Error and Cause	Corrective Action
"CAN'T EXECUTE (I)" (Checked at the occurrence of interruption)	15	Stop	(1) Although the interrupt module is used, there is no number of interrupt pointer I, which corresponds to that module, in the program or there are multiple numbers. (2) No IRET instruction has been entered in the interrupt program. (3) There is IRET instruction in other than the interrupt program.	(1) Check for the presence of interrupt program which corresponds to the interrupt unit and create an interrupt program or reduce the same numbers of I. (2) Check if there is IRET instruction in the interrupt program and enter the IRET instruction. (3) Check if there is IRET instruction in other than the interrupt program and delete the IRET instruction.
"CASSETTE ERROR" (Checked at power-on)	16	Stop	The memory cassette is not loaded.	Load the memory cassette and perform reset.
"RAM ERROR" (Checked at power-on)	20	Stop	The CPU has checked if write and read operations can be performed properly to the data memory area of CPU, and as a result, either or both has not been performed.	Since this is CPU hardware error, consult nearby service center, representative, or branch.
"OPE.CIRCUIT ERR." (Checked at power-on)	21	Stop	The operation circuit, which performs the sequence processing in the CPU, does not operate properly.	
"WDT ERROR" (Checked at the execution of END processing)	22	Stop	Scan time exceeds watch dog error monitor time. (1) Scan timer of user program has been exceeded. (2) Scan time has lengthened due to instantaneous power failure which occurred during scan.	(1) Calculate and check the scan time of user program and reduce the scan time by use of CJ instruction, etc. (2) Monitor the content of special register D9005 by use of peripheral equipment. When the content is other than 0, line voltage is insufficient. Therefore, check the power and reduce the fluctuation of voltage.
"SUB-CPU ERROR" (Checked continuously)	23 (During run) 26 (At power-on)	Stop	Sub-CPU is locked-up or defective.	Since this is CPU hardware error, consult nearby service center, representative, or branch.
NO' END' (Checked at the execution of END instruction)	24	Stop	(1) When the END instruction is executed, another instruction code has been read due to noise, etc. (2) The END instruction has changed to another instruction code for some reason.	Perform reset and run. If the same error is displayed again, it is the CPU hardware error. Therefore, consult nearby service center, representative, or branch.
"WDT ERROR" (Checked continuously)	25	Stop		Since this is CPU hardware error, consult nearby service center, representative, or branch.

Table 3.1 Error Code List (Continue)

Error Message	Content of Special Register D9008 (BIN value)	CPU States	Error and Cause	Corrective Action
"UNIT VERIFY ERR." (Checked continuously)	31	Run (Stop)	I/O module data are different from those at power-on. (1) The I/O module (including the special function module) is incorrectly loaded or has been removed, or a different unit has been loaded.	(1) Among special registers D9116 to D9123, the bit corresponding to the module of verify error is "1". Therefore, monitor the registers by use of peripheral equipment and check for the module with "1" and make replacement. (2) When the present unit arrangement is OK, perform reset with the reset switch.
"FUSE BREAK OFF" (Checked continuously)	32	Run (Stop)	A fuse is blown in an output module.	(1) Check the fuse blown indicator LED of output module and change the fuse of module of which LED is on. (2) The check of the defective unit can also be made by the peripheral equipment. Among special registers D9116 to D9123, the bit corresponding to the unit of verify error is "1". Therefore, make checks by monitoring the registers.
"WDT ERROR" (Checked continuously)	25	Stop	Hardware fault. (A3HCPU only)	The CPU is executing an endless loop. A1N, A2N, A3N, A3HCPU.
"CONTROL-BUS ERR." (Checked at the execution of FROM and TO instructions)	40	Stop	The FROM and TO instructions cannot be executed. (1) Error of control bus with special function module.	This error can be caused by a special function module, CPU module or base unit hardware. Therefore, change each module and check the defective module. For the defective module, consult nearby service center, representative, or branch.
"SP.UNIT DOWN" (Checked at the execution of FROM and TO instructions)	41	Stop	When the FROM or TO instruction is executed, access has been made to the special function module but the answer is not given. (1) The accessed special function module is defective.	Since this is an accessed special function module error, consult nearby service center, representative, or branch.
"LINK UNIT ERROR"	42	Stop	AJ71R22 or AJ71P22 is loaded in the master station.	Remove the AJ71R22 or PJ71P22 from the master station. After correction, perform reset and start at the initial operation.
"I/O INT. ERROR"	43	Stop	Although the interrupt module is not loaded, interruption has occurred.	This is a specific module hardware error. Therefore, change the unit and check the defective unit. For the defective unit, consult nearby service center, representative, or branch.

Table 3.1 Error Code List (Continue)

APP

Error Message	Content of Special Register D9008 (BIN value)	CPU States	Error and Cause	Corrective Action
"SP. UNIT LAY. ERROR."	44	Stop	(1) Three or more computer link units are loaded with respect to one CPU module. (2) Two or more units of AJ71P22 or AJ71R22 are loaded. (3) Two or more interrupt units are loaded.	(1) Reduce the computer link modules to two or less. (2) Reduce the AJ71P22 or AJ71R22 to one or less. (3) Reduce the interrupt module to one. (4) Re-set the I/O assignment of parameter setting by use of A6GPP, A6PHP, A6HGP according to the actually loaded special function module.
"SP. UNIT ERROR" (Checked at the execution of FROM and TO instructions)	46	Stop (Run)	Access (execution of FROM to TO instruction) has been made to a location where there is not special function unit.	Read the error step by use of peripheral equipment, and check and correct the content of FROM or TO instruction at that step by use of peripheral equipment.
"LINK P-MTR. ERROR"	47	Run	(1) The contents written to the parameter area of link by setting the link range in the parameter setting of A6GPP, A6PHP, A6HGP are different from the link parameter contents for some reason. (2) The setting of the total number of slave stations is 0.	(1) Write parameters again and make check. (2) When the error is displayed again, it is hardware error. Therefore, consult nearby service center, representative, or branch.
"OPERATION ERROR" (Checked during execution of instruction)	50	Run (Stop)	(1) The result of BCD conversion has exceeded the specified range (9999 or 99999999). (2) Operation impossible because specified device range has been exceeded. (3) File registers used in program without parameter setting.	Examine the program step indicated by the PC diagnostics and correct.
"MAIN CPU DOWN" (Interrupt fault)	60	Stop	(1) INT instruction processed in microcomputer program area. (2) CPU malfunction due to noise. (3) Hardware fault.	(1) Remove INT. (2) Eliminate noise. (3) Consult Mitsubishi representative.
"BATTERY ERROR" (Checked at power-on)	70	Run	(1) Battery voltage low. (2) Battery not connected.	Connect battery for RAM and/or power failure data back up.

Table 7.1 Error Code List

APPENDIX 4 ASCII Code Table

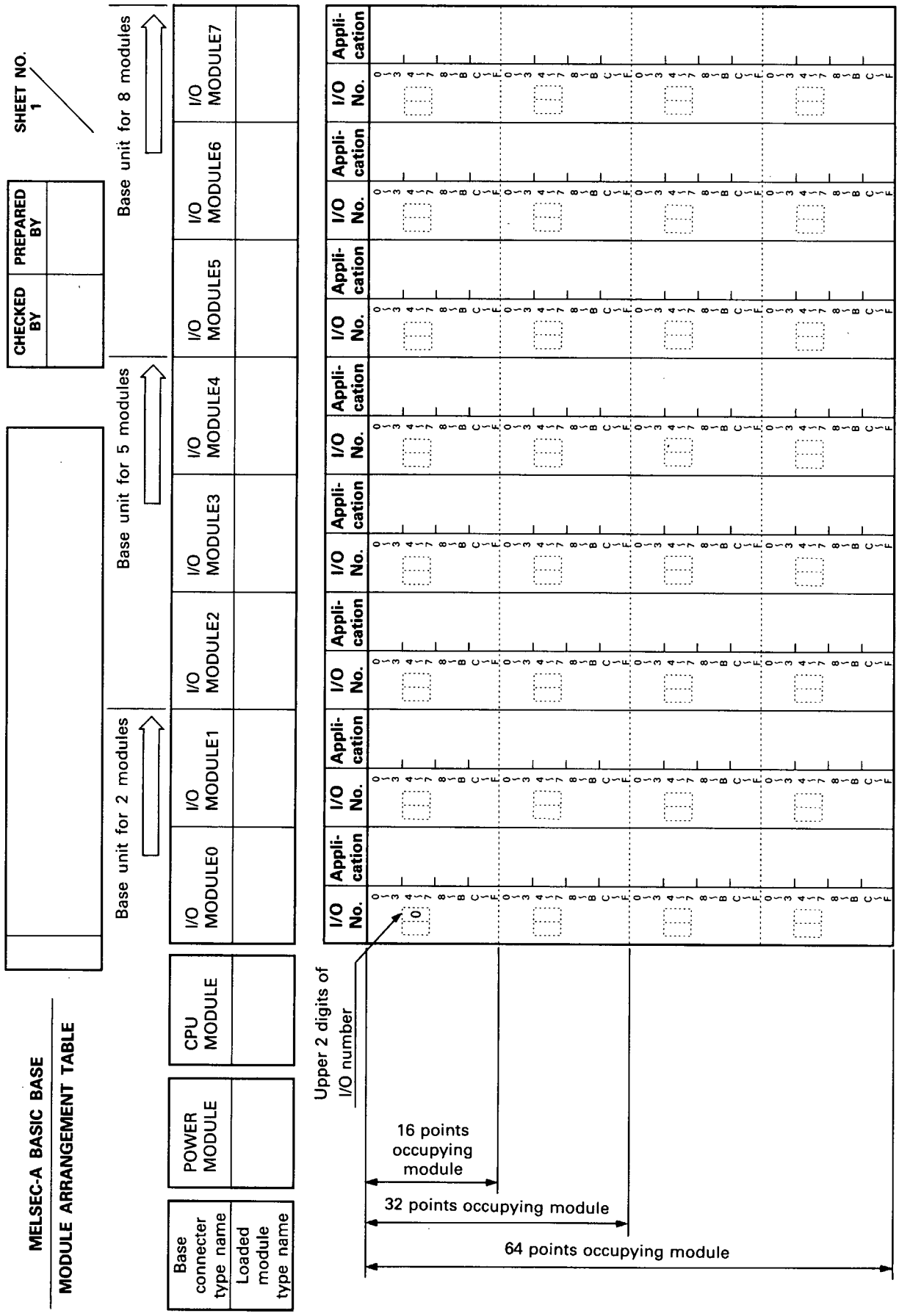
Bit number b7	b6	b5	b4	b3	b2	b1	Line \ Column	0	0	0	0	1	1	1	1
								0	0	1	1	0	0	1	1
								0	1	0	1	0	1	0	1
								0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	(TC7)DLE	SP	0	@	P	`	p
			0	0	0	1	1	(TC1)SOH	DC1	!	1	A	Q	a	q
			0	0	1	0	2	(TC2)STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	(TC3)ETX	DC3	#	3	C	S	c	s
			0	1	0	0	4	(TC4)EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	(TC5)ENQ	(TC8)NAK	%	5	E	U	e	u
			0	1	1	0	6	(TC6)ACK	(TC9)SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	(TC10)ETB	'	7	G	W	g	w
			1	0	0	0	8	FE0(BS)	CAN	(	8	H	X	h	x
			1	0	0	1	9	FE1(HT)	EM	)	9	I	Y	i	y
			1	0	1	0	10	FE2(LF/NL)	SUB	*	:	J	Z	j	z
			1	0	1	1	11	FE3(VT)	ESC	+	;	K	[	k	{
			1	1	0	0	12	FE4(FF)	IS4(FS)	,	<	L	\	l	|
			1	1	0	1	13	FE5(CR)	IS3(GS)	-	=	M	]	m	}
			1	1	1	0	14	SO	IS2(RS)	.	>	N	^	n	~
			1	1	1	1	15	SI	IS1(US)	/	?	O	_	o	DEL

ASCII Codes (Control codes)

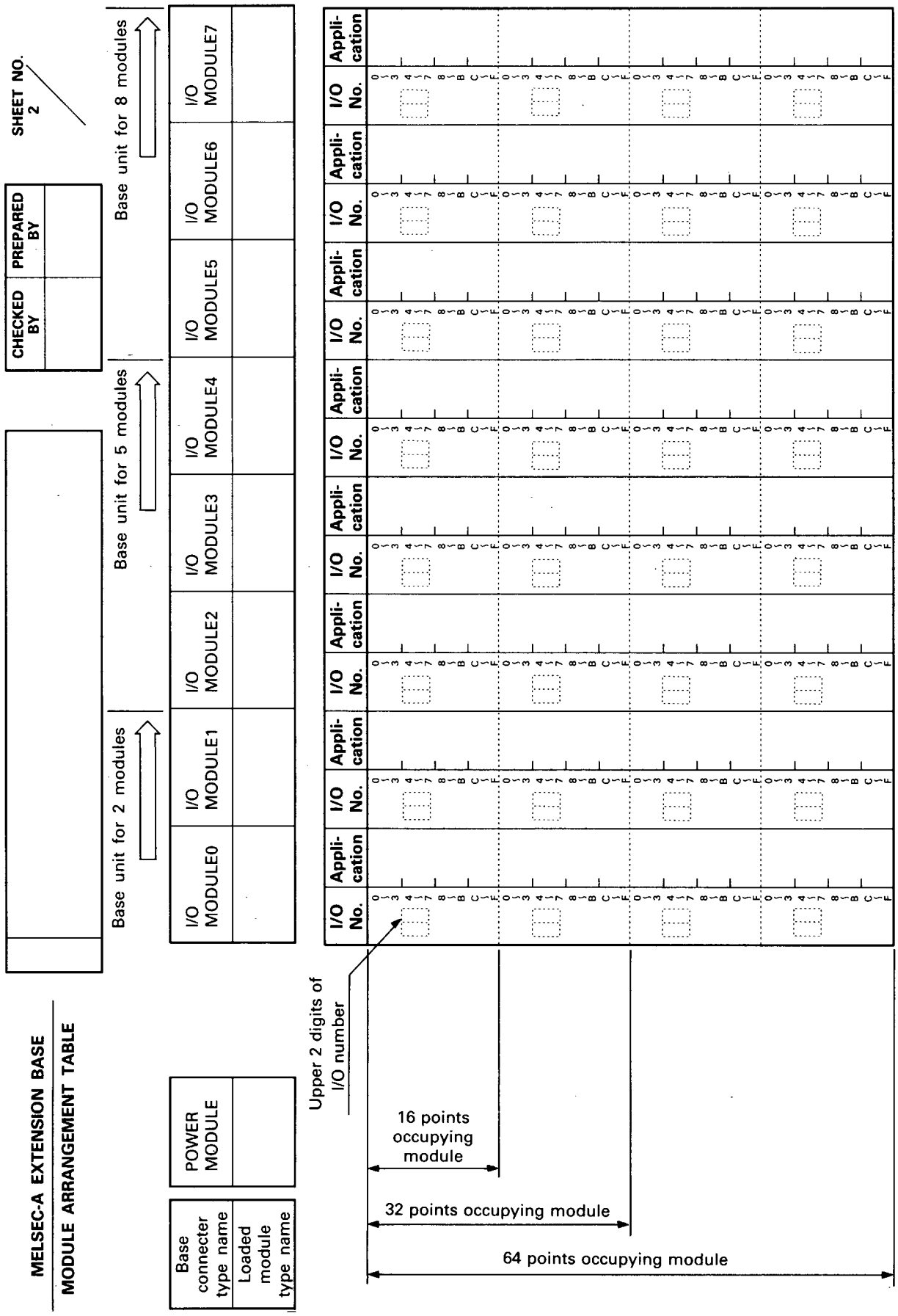
NUL	(Blank)
SOH	(Start of Heading)
STX	(Start of Text)
ETX	(End of Text)
EOT	(End of Transmission)
ENQ	(Enquiry)
ACK	(Acknowledge <Positive>)
BEL	(Bell)
BS	(Backspace)
HT	(Horizontal Tabulation)
LF	(Line Feed)
VT	(Vertical Tabulation)
FF	(Form Feed)
CR	(Carriage Return)
SO	(Shift Out)
SI	(Shift In)
DLE	(Data Link Escape)
DC1	(Device Control 1)
DC2	(Device Control 2)
DC3	(Device Control 3)
DC4	(Device Control 4-Stop)
NAK	(Negative Acknowledge)
SYN	(Synchronous Idle)
ETB	(End of Transmission Block)
CNA	(Cancel)
EM	(End of Medium)
SUB	(Substitute Character)
ESC	(Escape)
FS	(File Separator)
GS	(Group Separator)
RS	(Record Separator)
US	(Unit Separator)
SP	(Space)
DEL	(Delete/Rubout)

APPENDIX 5 Formats of Program Sheets

Sheet format 1-1



Sheet format 1-2



Sheet format 1-3

MELSEC-A

CODING SHEET

CHECKED BY	PREPARED BY	SHEET NO.

Step Number	Instruction	Device	Remarks
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			

Sheet format 1-4

MELSEC-A

BIT DEVICE LIST

CHECKED BY	PREPARED BY	SHEET NO.

	Signal	Description
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		

	Signal	Description
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

APP

Sheet format 1-5

MELSEC-A

WORD DEVICE LIST

CHECKED BY	PREPARED BY	SHEET NO.

	Data (16 bits/data)	Description
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

	Data (16 bits/data)	Description
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

APP

Sheet format 1-6

MELSEC-A

ANNUNCIATOR LIST

CHECKED BY	PREPARED BY	SHEET NO.

Failure Memory Number	External Failure Name	Failure Type, Condition ➡ Troubleshooting Point
F 0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
F 0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
F 0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

APP

Sheet format 1-7

MELSEC-A

TIMER, COUNTER LIST

		CHECKED BY	PREPARED BY	SHEET NO.

	Number	Set Value K				Description	Application, Operation (Count Input), etc.
	0						
	1						
	2						
	3						
	4						
	5						
	6						
	7						
	8						
	9						
	0						
	1						
	2						
	3						
	4						
	5						
	6						
	7						
	8						
	9						
	0						
	1						
	2						
	3						
	4						
	5						
	6						
	7						
	8						
	9						

APP

IB (NA) 66147-A

INDEX OF INSTRUCTIONS AND DEVICES

IMPORTANT

The components on the printed circuit boards will be damaged by static electricity, so avoid handling them directly. If it is necessary to handle them take the following precautions.

(1) Ground human body and work bench.

(2) Do not touch the conductive areas of the printed circuit board and its electrical parts with any non-grounded tools etc.

Under no circumstances will Mitsubishi Electric be liable or responsible for any consequential damage that may arise as a result of the installation or use of this equipment.

All examples and diagrams shown in this manual are intended only as an aid to understanding the text, not to guarantee operation. Mitsubishi Electric will accept no responsibility for actual use of the product based on these illustrative examples.

Owing to the very great variety in possible applications of this equipment, you must satisfy yourself as to its suitability for your specific application.

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE: MITSUBISHI DENKI BLDG MARUNOUCHI TOKYO 100 TELEX: J24532 CABLE MELCO TOKYO
NAGOYA WORKS : 1-14 , YADA-MINAMI 5 , HIGASHI-KU , NAGOYA , JAPAN

IB (NA) 66147-A (8809) MEE Printed in Japan